



SỞ GIÁO DỤC VÀ ĐÀO TẠO HÀ NỘI

GIÁO TRÌNH

Kỹ thuật số và mạch logic

DÙNG TRONG CÁC TRƯỜNG TRUNG HỌC CHUYÊN NGHIỆP



NHÀ XUẤT BẢN HÀ NỘI

SỞ GIÁO DỤC VÀ ĐÀO TẠO HÀ NỘI

KS. CHU KHẮC HUY (*Chủ biên*)

KS. NGUYỄN THỊ THU THỦY

GIÁO TRÌNH KỸ THUẬT SỐ VÀ MẠCH LOGIC

(Dùng trong các trường THCN)

NHÀ XUẤT BẢN HÀ NỘI - 2006

Lời giới thiệu

Nước ta đang bước vào thời kỳ công nghiệp hóa, hiện đại hóa nhằm đưa Việt Nam trở thành nước công nghiệp văn minh, hiện đại.

Trong sự nghiệp cách mạng to lớn đó, công tác đào tạo nhân lực luôn giữ vai trò quan trọng. Báo cáo Chính trị của Ban Chấp hành Trung ương Đảng Cộng sản Việt Nam tại Đại hội Đảng toàn quốc lần thứ IX đã chỉ rõ: “Phát triển giáo dục và đào tạo là một trong những động lực quan trọng thúc đẩy sự nghiệp công nghiệp hóa, hiện đại hóa, là điều kiện để phát triển nguồn lực con người - yếu tố cơ bản để phát triển xã hội, tăng trưởng kinh tế nhanh và bền vững”.

Quán triệt chủ trương, Nghị quyết của Đảng và Nhà nước và nhận thức đúng đắn về tầm quan trọng của chương trình, giáo trình đối với việc nâng cao chất lượng đào tạo, theo đề nghị của Sở Giáo dục và Đào tạo Hà Nội, ngày 23/9/2003, Ủyban nhân dân thành phố Hà Nội đã ra Quyết định số 5620/QĐ-UB cho phép Sở Giáo dục và Đào tạo thực hiện đề án biên soạn chương trình, giáo trình trong các trường Trung học chuyên nghiệp (THCN) Hà Nội. Quyết định này thể hiện sự quan tâm sâu sắc của Thành ủy, UBND thành phố trong việc nâng cao chất lượng đào tạo và phát triển nguồn nhân lực Thủ đô.

Trên cơ sở chương trình khung của Bộ Giáo dục và Đào tạo ban hành và những kinh nghiệm rút ra từ thực tế đào tạo, Sở Giáo dục và Đào tạo đã chỉ đạo các trường THCN tổ chức biên soạn chương trình, giáo trình một cách khoa học, hệ

thống và cập nhật những kiến thức thực tiễn phù hợp với đối tượng học sinh THCS Hà Nội.

Bộ giáo trình này là tài liệu giảng dạy và học tập trong các trường THCS ở Hà Nội, đồng thời là tài liệu tham khảo hữu ích cho các trường có đào tạo các ngành kỹ thuật - nghiệp vụ và đông đảo bạn đọc quan tâm đến vấn đề hướng nghiệp, dạy nghề.

Việc tổ chức biên soạn bộ chương trình, giáo trình này là một trong nhiều hoạt động thiết thực của ngành giáo dục và đào tạo Thủ đô để kỷ niệm “50 năm giải phóng Thủ đô”, “50 năm thành lập ngành” và hướng tới kỷ niệm “1000 năm Thăng Long - Hà Nội”.

Sở Giáo dục và Đào tạo Hà Nội chân thành cảm ơn Thành ủy, UBND, các sở, ban, ngành của Thành phố, Vụ Giáo dục chuyên nghiệp Bộ Giáo dục và Đào tạo, các nhà khoa học, các chuyên gia đầu ngành, các giảng viên, các nhà quản lý, các nhà doanh nghiệp đã tạo điều kiện giúp đỡ, đóng góp ý kiến, tham gia Hội đồng phản biện, Hội đồng thẩm định và Hội đồng nghiệm thu các chương trình, giáo trình.

Đây là lần đầu tiên Sở Giáo dục và Đào tạo Hà Nội tổ chức biên soạn chương trình, giáo trình. Dù đã hết sức cố gắng nhưng chắc chắn không tránh khỏi thiếu sót, bất cập. Chúng tôi mong nhận được những ý kiến đóng góp của bạn đọc để từng bước hoàn thiện bộ giáo trình trong các lần tái bản sau.

GIÁM ĐỐC SỞ GIÁO DỤC VÀ ĐÀO TẠO

Lời nói đầu

Cùng với sự phát triển mạnh mẽ của các ngành kỹ thuật khác, kỹ thuật điện tử cũng có những bước tiến nhảy vọt trong những năm gần đây, đặc biệt là kỹ thuật điện tử số. Hiện nay các thiết bị số đã có mặt trong nhiều lĩnh vực. Vì vậy để hiểu, vận hành và sửa chữa được các thiết bị số, cần đào tạo được đội ngũ cán bộ kỹ thuật có kiến thức vững vàng về điện tử số.

Nhằm phục vụ cho việc dạy và học ở các trường Trung học chuyên nghiệp môn kỹ thuật số và mạch logic, chúng tôi đã tiến hành biên soạn giáo trình Kỹ thuật số và mạch logic. Giáo trình gồm 11 chương như sau:

Chương 1. Hệ thống đếm và mã

Chương 2. Đại số logic

Chương 3. Bộ cộng, trừ và so sánh

Chương 4. Bộ dồn kênh và phân kênh

Chương 5. Các bộ mã hoá và giải mã

Chương 6. Các phần tử nhớ cơ bản

Chương 7. Bộ đếm

Chương 8. Bộ ghi dịch

Chương 9. Bộ nhớ

Chương 10. Chuyển đổi tín hiệu

Chương 11. Các vi mạch số thông dụng và một số sơ đồ thực tế

Trong mỗi chương chúng tôi đã cố gắng đưa những nội dung và lượng kiến thức phù hợp với đối tượng sử dụng. Phân tích kỹ những mạch cơ bản và đưa ra những gợi ý sâu sắc cho học sinh trong phần câu hỏi và bài tập cuối chương.

Trong quá trình biên soạn giáo trình, chúng tôi đã nhận được sự đóng góp ý kiến nhiệt tình, sâu sắc của các nhà chuyên môn, các giảng viên giảng dạy ở các trường Đại học, các đồng nghiệp đang công tác tại trường với chúng tôi và đặc biệt là Tiến sĩ Nguyễn Văn Thuấn, giảng viên Học viện Kỹ thuật Quân sự, Thạc sĩ Dương Văn Phương và Thạc sĩ Nguyễn Hoàng Dũng, giảng viên của trường Đại học Bách khoa Hà Nội. Chúng tôi xin chân thành cảm ơn.

Mặc dù đã cố gắng, song trong quá trình biên soạn chắc chắn không tránh khỏi những sai sót. Mong nhận được những ý kiến đóng góp phê bình của bạn đọc.

TÁC GIẢ

Bài mở đầu

ĐỐI TƯỢNG, NỘI DUNG VÀ PHƯƠNG PHÁP NGHIÊN CỨU MÔN HỌC

1. Đối tượng của môn học

Kỹ thuật số và mạch logic là môn học nghiên cứu những kiến thức cơ bản về các mạch tổ hợp logic và mạch tuần tự. Trên cơ sở đó, giúp cho người cán bộ kỹ thuật có sự hiểu biết đầy đủ về nguyên lý làm việc của các mạch điện tử số như các mạch cộng, trừ, so sánh, mạch dồn kênh, phân kênh, mạch mã hóa, giải mã, các loại mạch đếm, mạch ghi dịch, bộ nhớ và các mạch chuyển đổi tín hiệu tương tự sang số và số sang tương tự. Đây là cơ sở để người cán bộ kỹ thuật tiếp cận vào các thiết bị điện tử số.

2. Nội dung của môn học

Giáo trình *Kỹ thuật số và mạch logic* ngoài việc đề cập tới những nội dung cơ bản về mạch số như sơ đồ, nguyên lý hoạt động của các mạch, còn giới thiệu một loạt các kiến thức khác như phương pháp thiết kế mạch tổ hợp, một số mạch số ứng dụng trong thực tế. Với nội dung như vậy, giáo trình không những giúp cho người học có được kiến thức để tiếp cận các môn học chuyên ngành, có khả năng khai thác, sử dụng và khắc phục được những sự cố trong các thiết bị số mà còn là tài liệu tham khảo bổ ích cho đội ngũ cán bộ kỹ thuật đang làm việc trong các lĩnh vực có liên quan đến kỹ thuật điện tử số.

Môn học được bố trí thành 11 chương, nhằm cung cấp cho học sinh những kiến thức:

- Nguyên lý hoạt động của các phần tử cơ bản trong kỹ thuật số như các cổng logic, các mạch tổ hợp, các mạch dây, cấu trúc và hoạt động của bộ nhớ ROM, RAM và các kiểu chuyển đổi tín hiệu tương tự sang số, số sang tương tự.

- Phương pháp thiết kế được các mạch tổ hợp đơn giản như các mạch mã hoá, giải mã, dồn kênh, phân kênh.

- Phân tích đặc điểm và hoạt động của các mạch ứng dụng trong thực tế như mạch tạo xung, mạch chia tần, mạch đếm có hiển thị kết quả đếm bằng đèn 7 thanh, mạch điều khiển các đối tượng thông qua xử lý số v.v.

Qua môn học, rèn luyện cho học sinh ý thức tích cực sáng tạo, khả năng tư duy logic, nhiệt tình và yêu thích nghề nghiệp.

3. Phương pháp nghiên cứu môn học

Kỹ thuật số và mạch logic rất cần cho các cán bộ kỹ thuật và người lao động khi làm các công việc liên quan đến kỹ thuật điện tử. Đây là môn học bắt buộc đối với học sinh theo học các ngành, nghề liên quan đến kỹ thuật điện tử. Để có thể nắm bắt được nội dung của môn học, học sinh cần phải có các kiến thức của các môn khác như Vật liệu điện tử, Kỹ thuật mạch điện tử và Thực hành điện tử cơ bản... Trong quá trình học tập, học sinh cần phải kết hợp giữa học lý thuyết ở trên lớp với việc làm các bài tập. Trước khi làm bài tập cần nắm vững kiến thức lý thuyết cơ bản. Ngoài kiến thức được nêu trong giáo trình, người học cần phải thường xuyên cập nhật các thông tin mới thông qua các giáo trình tham khảo. Các kiến thức trình bày trong giáo trình mặc dù chỉ là kiến thức cơ bản, song để có thể lĩnh hội được nhanh chóng thì người học cần phải tuân thủ theo kết cấu của giáo trình và cần có sự hướng dẫn của các giáo viên chuyên ngành.

Tóm lại, để có thể học tốt môn học, người học cần phải xác định rõ mục đích và yêu cầu của môn học. Luôn luôn kết hợp chặt chẽ giữa học lý thuyết với học thực hành. Đồng thời tích cực ôn luyện theo sự hướng dẫn của các giáo viên, đặc biệt là ghi nhớ những kết luận rút ra từ các kết quả thực hành, thực tập trên các thiết bị thật hoặc trên các mô hình đã có sẵn trong các phòng thực hành.

Chương 1

HỆ THỐNG ĐẾM VÀ MÃ

Mục tiêu

- Hiểu và làm được các bài tập về chuyển đổi số giữa các hệ đếm, chuyển đổi giữa các mã.

- Thực hiện thành thạo các phép toán cộng, trừ ở hệ đếm 2, hệ đếm 8 và hệ đếm 16.
- Rèn luyện tính tư duy logic và sáng tạo trong công việc chuyên môn.

Nội dung trọng tâm

- Giới thiệu về các hệ đếm và các mã thông dụng trong kỹ thuật số.
- Các phương pháp chuyển đổi số giữa các hệ đếm và chuyển đổi các mã.
- Phương pháp thực hiện các phép toán cộng và phép toán trừ trong hệ đếm hai, tám và mười sáu.

I. BIỂU DIỄN SỐ TRONG CÁC HỆ THỐNG ĐẾM

1. Các khái niệm cơ bản

Đối với các thiết bị số (*digital*), việc xử lý thông tin được thực hiện thông qua những con số đặc biệt, biểu diễn dưới những dạng mã khác nhau, nằm trong các *hệ thống đếm* khác nhau. Để hiểu rõ vấn đề hơn, ta xét một số khái niệm cơ bản sau đây:

+ Hệ thống đếm: Hệ thống đếm là tổ hợp các quy tắc gọi và biểu diễn các con số có giá trị xác định.

+ Chữ số: Chữ số là những ký hiệu dùng để biểu diễn một con số.

+ Phân loại hệ thống đếm: Tùy theo phương pháp biểu diễn các con số mà hệ thống đếm có thể chia ra thành hai loại là *hệ thống đếm theo vị trí* và *hệ thống đếm không theo vị trí*.

- Hệ thống đếm theo vị trí là hệ thống đếm mà trong đó giá trị về mặt số lượng của mỗi chữ số phụ thuộc vào vị trí của chữ số đó nằm trong con số.

Ví dụ: Trong hệ thống đếm mười (thập phân) theo chữ số Ả rập thông thường, xét chữ số 4 ở hai con số sau, rõ ràng là chúng có giá trị khác nhau:

1234: chữ số 4 biểu diễn giá trị 4 đơn vị.

4321: chữ số 4 biểu diễn giá trị $4 \cdot 10^3$ đơn vị.

- Hệ thống đếm không theo vị trí là hệ thống đếm mà giá trị về mặt số lượng của chữ số không phụ thuộc vào vị trí của nó nằm trong con số.

Ví dụ: Trong hệ thống đếm theo chữ số La mã, xét hai con số sau:

XXIII = 23 đơn vị

XXXIX = 39 đơn vị

Rõ ràng là giá trị của các chữ số X ở cả hai con số trên không phụ thuộc vào vị trí của nó nằm trong con số.

Hệ thống đếm không theo vị trí (mà tiêu biểu là hệ thống đếm theo chữ số La mã) trở nên cồng kềnh khi cần biểu diễn các con số có giá trị lớn, do đó hiện nay người ta ít dùng.

Để thuận tiện, từ đây trở đi trong toàn bộ cuốn sách này chúng tôi gọi tắt một hệ thống đếm theo vị trí là một *hệ đếm* và trong nhiều trường hợp gọi tắt là *hệ* như "hệ hai", "hệ mười", v.v...

2. Các hệ đếm thông dụng

2.1. Hệ mười

Hệ mười là hệ đếm thường dùng nhất trong sinh hoạt và công việc của chúng ta. Trong hệ đếm này có 10 chữ số, được ký hiệu là 0, 1, 2, 3, 4, 5, 6, 7, 8 và 9 để mã hoá số 0 và 9 số tự nhiên đầu tiên. Cơ số của hệ đếm là 10. Từ số lớn hơn 9, nhờ cách ghi số theo vị trí, trong đó số có vị trí bất kỳ có trọng số gấp 10 lần số có vị trí bên phải liền kề. Ta có thể dùng 10 chữ số trong hệ đếm để biểu diễn mọi con số.

Ví dụ: $243,65 = 2 \times 10^2 + 4 \times 10^1 + 3 \times 10^0 + 6 \times 10^{-1} + 5 \times 10^{-2}$

hay $901,78 = 9 \times 10^2 + 0 \times 10^1 + 1 \times 10^0 + 7 \times 10^{-1} + 8 \times 10^{-2}$

2.2. Hệ hai

Hệ hai được dùng rộng rãi nhất trong mạch số. Trong hệ hai, mỗi vị trí số chỉ có hai khả năng lấy giá trị là 0 và 1. Cơ số đếm của hệ hai là 2. Ví dụ về việc biểu diễn các số ở hệ hai như sau:

$101,11 = 1 \times 2^2 + 0 \times 2^1 + 1 \times 2^0 + 1 \times 2^{-1} + 1 \times 2^{-2}$

$01100,010 = 1 \times 2^3 + 1 \times 2^2 + 0 \times 2^1 + 0 \times 2^0 + 0 \times 2^{-1} + 1 \times 2^{-2}$

2.3. Hệ tám

Hệ tám, có tám ký hiệu như ở hệ mười, bắt đầu từ ký hiệu 0 và kết thúc là ký hiệu 7 để biểu diễn các số ở hệ đếm này. Cơ số của hệ đếm là 8. Ví dụ về biểu diễn số ở hệ tám như sau:

$$513,26 = 5 \times 8^2 + 1 \times 8^1 + 3 \times 8^0 + 2 \times 8^{-1} + 6 \times 8^{-2}$$

$$\text{hoặc } 703,41 = 7 \times 8^2 + 0 \times 8^1 + 3 \times 8^0 + 4 \times 8^{-1} + 1 \times 8^{-2}$$

2.4. Hệ mười sáu

Ở hệ đếm này sử dụng 16 ký hiệu để biểu diễn 16 con số của hệ đếm. Từ giá trị 1 đến 9 biểu diễn như ở hệ đếm mười, từ 10 đến 15, mỗi số được ký hiệu bằng một chữ cái tương ứng A, B, C, D, E và F.

Ví dụ về biểu diễn số ở hệ mười sáu: 3C2; 4FF; 516,4A.

Tương tự như các hệ đếm trên, số 516,4A trong hệ đếm mười sáu chính là:

$$(516,4A)_{16} = 5 \times 16^2 + 1 \times 16^1 + 6 \times 16^0 + 4 \times 16^{-1} + 10 \times 16^{-2}.$$

II. CHUYỂN ĐỔI SỐ GIỮA CÁC HỆ ĐẾM

1. Chuyển các số từ hệ mười sang hệ hai và ngược lại

1.1. Chuyển từ hệ mười sang hệ hai

Để chuyển số từ hệ mười sang hệ hai ta phải tiến hành theo hai bước: chuyển đổi phần trước dấu phẩy và chuyển đổi phần sau dấu phẩy (phần lẻ) theo hai quy tắc khác nhau.

1.1.1. Chuyển các số trước dấu phẩy

Quy tắc chuyển đổi phần trước dấu phẩy: Chia liên tiếp số hệ mười và các thương số của mỗi lần chia cho 2, tới khi kết quả chia bằng 0 thì dừng. Phần dư của mỗi lần chia chính là giá trị các bit của số hệ hai cần tìm, bắt đầu từ bit có trọng số nhỏ nhất tới bit có trọng số lớn nhất.

Ví dụ 1: Chuyển đổi số 21 hệ mười sang hệ hai, ta làm như sau:

$$21 : 2 = 10 \text{ dư } 1 \text{ (số dư 1 ở đây là bit có trọng số nhỏ nhất)}$$

$$10 : 2 = 5 \text{ dư } 0$$

$$5 : 2 = 2 \text{ dư } 1$$

$$2 : 2 = 1 \text{ dư } 0$$

$$1 : 2 = 0 \text{ dư } 1 \text{ (số dư 1 ở đây là bit có trọng số lớn nhất).}$$

$$\Rightarrow (21)_{10} = (10101)_2$$

Ví dụ 2: Chuyển đổi số 37 ở hệ mười sang hệ hai, ta thực hiện như sau:

$$37 : 2 = 18 \text{ dư } 1$$

$$18 : 2 = 9 \text{ dư } 0$$

$$9 : 2 = 4 \text{ dư } 1$$

$$4 : 2 = 2 \text{ dư } 0$$

$$2 : 2 = 1 \text{ dư } 0$$

$$1 : 2 = 0 \text{ dư } 1. \text{ Số dư từ dưới lên chính là số hệ hai cần tìm.}$$

Như vậy $(37)_{10} = (100101)_2$.

1.1.2. Chuyển các số sau dấu phẩy

Quy tắc chuyển đổi phần sau dấu phẩy: Nhân liên tiếp phần lẻ của số cần chuyển đổi và các tích tìm được với hai cho tới khi tích tìm được không còn phần lẻ thì dừng. (Có trường hợp khi nhân liên tiếp nhiều lần mà vẫn không hết phần lẻ ta có thể vẫn dừng và chỉ lấy một số chữ số nhất định để làm số sau dấu phẩy cho hệ hai). Số ở phần nguyên của các tích trong mỗi lần nhân chính là giá trị các bit của số nhị phân cần tìm, bắt đầu từ bit có trọng số lớn nhất tới bit có trọng số nhỏ nhất.

Ví dụ 1: Chuyển đổi số 0,25 ở hệ 10 sang hệ 2 ta làm như sau:

$$0,25 \times 2 = 0,5 \quad \Rightarrow (0,25)_{10} = (0,01)_2$$

$$0,5 \times 2 = 1,0$$

Ví dụ 2: Chuyển số 0,62 ở hệ mười sang hệ hai như sau:

$$0,62 \times 2 = 1,24$$

$$0,24 \times 2 = 0,48$$

$$0,48 \times 2 = 0,96$$

đến đây ta có thể viết $(0,62)_{10} = (0,100)_2$ vẫn được, việc này phụ thuộc vào yêu cầu độ chính xác của công việc xử lý số liệu.

Ví dụ 3: $(83,70)_{10} = (?)_2$

Để chuyển đổi được các số có cả phần trước và sau dấu phẩy ta tách phần trước dấu phẩy và phần sau dấu phẩy ra làm 2 phần, sau đó áp dụng quy tắc chuyển đổi của từng phần. Kết quả chung có được bằng cách ghép hai kết quả riêng.

Áp dụng quy tắc chuyển đổi ta có:

$$(83)_{10} = (1010011)_2 \text{ và } (0,70)_{10} = (101)_2$$

$$\Rightarrow (83,70)_{10} = (1010011, 101)_2.$$

1.2. Chuyển các số từ hệ hai sang hệ mười

Ở phần chuyển đổi này cũng phải thực hiện theo hai bước, chuyển đổi phần trước dấu phẩy và phần sau dấu phẩy.

1.2.1. Chuyển các số trước dấu phẩy

Quy tắc: Từ số hệ hai cần chuyển đổi, viết mỗi chữ số của số cần chuyển đổi ở dạng tích gồm chữ số hệ hai với 2^i , số mũ i tương ứng với trọng số của chữ số hệ hai trong số cần chuyển đổi. Kết quả là tổng của các tích trên.

Ví dụ 1: Để chuyển đổi số hệ hai 1011 sang số hệ mười ta làm như sau:

1	0	1	1
$\rightarrow$		$\rightarrow$	$\rightarrow$
			$1.2^0 = 1$
			$1.2^1 = 2$
			$0.2^2 = 0$
			$1.2^3 = 8$

Kết quả là tổng của các số 1, 2, 0, 8 và bằng $(11)_{10}$

Ví dụ 2: $(11001)_2 = 1.2^4 + 1.2^3 + 0.2^2 + 0.2^1 + 1.2^0 = 16 + 8 + 1 = (25)_{10}$

1.2.2. Chuyển các số sau dấu phẩy

Quy tắc: Để chuyển số hệ hai sau dấu phẩy sang hệ mười ta tiến hành như sau, lấy từng chữ số sau dấu phẩy nhân với 2^{-i} , i là vị trí của các chữ số hệ hai sau dấu phẩy, i bắt đầu = 1 và tăng dần từ trái sang phải. Kết quả là tổng của các tích trên.

Ví dụ 1: Chuyển số hệ hai là 0,101 sang hệ mười

0,101	
$\rightarrow$	$\rightarrow$
	$1 \times 2^{-1} = 1/2 = 0,5$
	$0 \times 2^{-2} = 0 = 0$
	$1 \times 2^{-3} = 1/8 = 0,125$

Như vậy, số ở hệ mười là tổng của các số 0,5; 0 và 0,125 và bằng 0,625.

Ví dụ 2: $(0,110)_2 = (?)_{10}$

Thực hiện: $(0,110)_2 = 1.2^{-1} + 1.2^{-2} = 0,5 + 0,25 = (0,75)_{10}$

2. Chuyển các số từ hệ mười sang hệ tám và ngược lại

2.1. Chuyển các số từ hệ mười sang hệ tám

2.1.1. Chuyển các số trước dấu phẩy

Cũng tiến hành tương tự như chuyển từ hệ mười sang hệ hai nhưng khác ở chỗ ta chia số cần chuyển đổi cho 8 chứ không phải là 2.

Ví dụ 1: Chuyển số 93 hệ mười sang hệ tám:

$$\begin{array}{rcl} 93 : 8 & = & 11 \text{ dư } 5 \\ 11 : 8 & = & 1 \text{ dư } 3 \\ 1 : 8 & = & 0 \text{ dư } 1 \end{array} \quad \begin{array}{c} \uparrow \\ \Rightarrow (93)_{10} = (135)_8 \end{array}$$

Ví dụ 2: $(117)_{10} = (?)_8$

Thực hiện chuyển đổi:

$$\begin{array}{rcl} 117 : 8 & = & 14 \text{ dư } 5 \\ 14 : 8 & = & 1 \text{ dư } 6 \\ 1 : 8 & = & 0 \text{ dư } 1 \end{array}$$

$$\text{Vậy } (117)_{10} = (165)_8$$

2.1.2. Chuyển các số sau dấu phẩy

Cũng tương tự như chuyển đổi các số sau dấu phẩy từ hệ mười sang hệ hai. Chỉ có điều là chúng ta phải thay đổi cơ số hệ đếm hai bằng hệ đếm tám.

Ví dụ 1: Chuyển đổi số 0,46 ở hệ mười sang hệ tám ta làm như sau:

$$\begin{array}{rcl} 0,46 \times 8 & = & 3,68 \\ 0,68 \times 8 & = & 5,44 \\ 0,44 \times 8 & = & 3,52 \dots \end{array} \quad \downarrow$$

Kết quả có được số ở hệ tám là 0,353 (đọc số nguyên của tích theo chiều từ trên xuống).

Ví dụ 2: $(0,71)_{10} = (?)_8$

Thực hiện:

$$\begin{array}{rcl} 0,71 \times 8 & = & 5,68 \\ 0,68 \times 8 & = & 5,44 \\ 0,44 \times 8 & = & 3,52 \end{array}$$

$$\text{Kết quả: } (0,71)_{10} = (0,553)_8$$

2.2. Chuyển từ hệ tám sang hệ mười

Tương tự như chuyển từ hệ hai sang hệ mười chỉ cần thay cơ số hệ hai bằng cơ số cần chuyển đổi là hệ tám.

Ví dụ chuyển số 3207 hệ tám sang hệ mười, ta làm như sau:

$$\begin{array}{l}
 3207 \\
 \begin{array}{l}
 \rightarrow 7.8^0 = 7 \\
 \rightarrow 0.8^1 = 0 \\
 \rightarrow 2.8^2 = 128 \\
 \rightarrow 3.8^3 = 1536
 \end{array}
 \end{array}$$

Như vậy, kết quả số ở hệ mười là $7 + 128 + 1536 = 1671$.

3. Chuyển các số từ hệ mười sang hệ mười sáu và ngược lại

Hoàn toàn tương tự như chuyển đổi hệ mười sang hệ hai và ngược lại từ hệ hai sang hệ mười, chỉ cần lưu ý đổi cơ số của hệ đếm cho phù hợp.

Ví dụ 1: Chuyển số 67,45 ở hệ mười sang hệ mười sáu.

+ Chuyển phần nguyên:

$$67 : 16 = 4 \text{ dư } 3$$

$$4 : 16 = 0 \text{ dư } 4$$

$$(67)_{10} = (43)_{16}$$

+ Chuyển phần sau dấu phẩy:

$$0,45 \times 16 = 7,2$$

$$0,2 \times 16 = 3,2$$

$$(0,45)_{10} = (0,73)_{16}$$

$$\text{Vậy } (67,45)_{10} = (43,73)_{16}$$

Ví dụ 2: Chuyển số $(3E,8A)_{16}$ sang hệ đếm mười. Ta thực hiện như sau:

+ Chuyển phần trước dấu phẩy sang hệ mười:

$$(3E)_{16} = 3 \times 16^1 + 14 \times 16^0 = 48 + 14 = (62)_{10}$$

+ Chuyển phần sau dấu phẩy sang hệ mười:

$$(0,8A)_{16} = 8 \times 16^{-1} + 10 \times 16^{-2} = 0,5 + 0,039 = (0,539)_{10}$$

$$\Rightarrow (3E,8A)_{16} = (62,539)_{10}$$

Bảng 1.1. Biểu diễn số hệ tám theo hệ hai

Số ở hệ tám	Biểu diễn theo hệ hai
0	000
1	001
2	010

3	011
4	100
5	101
6	110
7	111

4. Chuyển các số từ hệ tám sang hệ hai và ngược lại

Hệ tám có 8 chữ số từ 0 đến 7, nếu dùng 3 cột số ở hệ hai ta sẽ biểu diễn được tất cả 8 chữ số hệ tám như bảng 1.1.

Nhìn vào bảng 1.1 ta thấy đối với số 2 và số 3 hệ tám chỉ cần 2 cột số hệ hai là biểu diễn đủ giá trị của nó hay như số 0 và số 1 chỉ cần một cột số là biểu diễn đủ, nhưng để chuẩn hoá toàn bộ 8 chữ số của hệ đếm tám nên người ta thêm các chữ số 0 vào đằng trước cho đủ 3 cột số ở hệ hai. Dựa vào bảng trên ta có phương pháp chuyển đổi các số giữa hệ tám và hệ hai như sau:

4.1. Chuyển từ hệ tám sang hệ hai

Quy tắc chuyển đổi: Mỗi chữ số hệ tám được viết bằng một tổ hợp 3 chữ số hệ hai tương ứng ở bảng 1.1.

- Ví dụ 1: $(213)_8 = (010\ 001\ 011)_2$; $(704)_8 = (111\ 000\ 100)_2$

- Ví dụ 2: $(502,371)_8 = (101\ 000\ 010, 011\ 111\ 001)_2$

4.2. Chuyển từ hệ hai sang hệ tám

* Đối với các số trước dấu phẩy người ta nhóm 3 chữ số nhị phân liên tiếp lại và đổi thành một chữ số hệ tám tương ứng như ở bảng 1.1, bắt đầu tính từ phải sang trái.

- Ví dụ 1: $(100111010)_2 = (100\ 111\ 010)_2 = (472)_8$

- Ví dụ 2: $(011101000)_2 = (011\ 101\ 000)_2 = (350)_8$

* Đối với các số sau dấu phẩy, cũng nhóm 3 chữ số nhị phân liên tiếp để đổi thành một chữ số hệ tám tương ứng như bảng 1.1. Tuy nhiên lúc này cần nhóm từ trái sang phải.

Ví dụ 1: $(0,110\ 010)_2 = (0,62)_8$

Ví dụ 2: $(101\ 110\ 000, 000\ 111\ 100)_2 = (560,074)_8$

5. Chuyển các số từ hệ mười sáu sang hệ hai và ngược lại

Hệ mười sáu có 16 chữ số, ký hiệu từ 0 đến F, nếu dùng 4 cột số ở hệ hai ta sẽ biểu diễn được tất cả 16 chữ số hệ mười sáu như bảng 1.2 sau:

Bảng 1.2. Biểu diễn số hệ mười sáu theo hệ hai

Số ở hệ mười sáu	Biểu diễn theo hệ hai
0	0000
1	0001
2	0010
3	0011
4	0100
5	0101
6	0110
7	0111
8	1000
9	1001
A	1010
B	1011
C	1100
D	1101
E	1110
F	1111

5.1. Chuyển các số từ hệ mười sáu sang hệ hai

Quy tắc chuyển đổi: Mỗi chữ số hệ mười sáu được viết bằng một tổ hợp 4 chữ số hệ hai tương ứng như bảng 1.2.

- Ví dụ :
- $(56C)_{16} = (0101\ 0110\ 1100)_2$
 - $(0,3F)_{16} = (0,0011\ 1111)_2$
 - $(9A1,04)_2 \approx (1001\ 1010\ 0001, 0000\ 0100)_2$

5.2. Chuyển các số từ hệ hai sang hệ mười sáu

Để chuyển đổi được các số ở hệ hai sang hệ mười sáu người ta tiến hành nhóm 4 chữ số nhị phân liên tiếp trong số cần chuyển đổi để đổi thành một chữ số ở hệ mười sáu. Cũng như ở hệ tám, đối với những số đứng trước dấu phẩy

được nhóm từ phải sang trái, còn đối với những số sau dấu phẩy thì được nhóm từ trái sang phải.

Ví dụ:

- $(1100\ 0010\ 0001)_2 = (C21)_{16}$
- $(0,0101\ 0001\ 1101)_2 = (0,51D)_{16}$
- $(1110\ 0101\ 0111, 1011\ 0011)_2 = (E57, B3)_{16}$

III. PHÉP TOÁN CỘNG, TRỪ TRONG CÁC HỆ ĐẾM HAI, TÁM VÀ MƯỜI SÁU

1. Phép toán cộng, trừ trong hệ đếm hai

1.1. Phép toán cộng trong hệ đếm hai

Tương tự như phép cộng 2 số trong hệ đếm mười mà chúng ta đã quen thuộc, phép cộng hai số hệ hai cũng bắt đầu từ cột số có trọng số nhỏ nhất.

Quy tắc thực hiện: $0 + 0 = 0$

$$0 + 1 = 1$$

$$1 + 0 = 1$$

$$1 + 1 = 0 \text{ và nhớ } 1.$$

Các ví dụ:

Ví dụ 1: $1001 + 110$ ta thực hiện:

$$\begin{array}{r} 1001 \\ + 110 \\ \hline 1111 \end{array}$$

Ở phép cộng này chúng ta thấy không có số nhớ nào xuất hiện khi cộng các cột số tương ứng với nhau.

Ví dụ 2: $1101 + 101$ ta thực hiện:

$$\begin{array}{r} 1101 \\ + 101 \\ \hline 10010 \end{array}$$

Ở ví dụ 2, khi thực hiện phép cộng ta thấy ngay cột thứ nhất (cột có trọng số thấp nhất) đã xuất hiện số nhớ. Khi đó cột thứ nhất của hàng kết quả nhận được giá trị bằng 0 đồng thời nhớ một sang cột có trọng số cao hơn ngay bên cạnh.

1.2. Phép toán trừ trong hệ đếm hai

* Quy tắc thực hiện: $0 - 0 = 0$

$$10 - 1 = 1 \text{ và đã vay } 1 \text{ ở cột cao hơn liền kề.}$$

$$1 - 0 = 1$$

$$1 - 1 = 0$$

* Các ví dụ:

$$\begin{array}{r} \text{Ví dụ 1:} \quad 1110 \\ -100 \\ \hline 1010 \end{array}$$

Ví dụ 2:

$$\begin{array}{r} 1100 \\ -111 \\ \hline 101 \end{array}$$

Trong ví dụ 2 này, khi thực hiện phép trừ ta thấy cần phải mượn số bị trừ ở cột cao liền kề để thực hiện phép trừ. Sau đó cũng như ở hệ mười, ta phải thực hiện trả lại cột cao giá trị đã mượn.

2. Phép toán cộng, trừ trong hệ đếm tám

2.1. Phép toán cộng trong hệ đếm tám

* Khi thực hiện phép cộng 2 số trong hệ đếm tám ta cũng tiến hành cộng các cột có cùng trọng số với nhau. Nếu kết quả ở một cột bất kỳ nào đó lại lớn hơn hoặc bằng tám thì ta thực hiện như sau:

- Lấy kết quả ở cột đó trừ cho 8.
- Hiệu số của phép trừ trên được ghi vào cột tương ứng của hàng kết quả.
- Cộng 1 vào cột có trọng số cao hơn ngay cạnh cột vừa thực hiện phép cộng có kết quả lớn hơn hoặc bằng tám.

* Một số ví dụ:

$$\begin{array}{r} \text{Ví dụ 1:} \quad 2301 \\ + 0145 \\ \hline 2446 \end{array}$$

$$\begin{array}{r} \text{Ví dụ 2:} \quad 4627 \\ + 1066 \\ \hline 5715 \end{array}$$

Ở ví dụ 2 khi cộng ở cột có trọng số 8^0 , $(6 + 7)$ được kết quả là 13 hệ 10. Ta lấy $13 - 8 = 5$. Viết 5 xuống hàng kết quả sau đó cộng 1 vào cột có trọng số 8^1

2.2. Phép toán trừ trong hệ đếm tám

* Phép toán trừ trong hệ đếm tám được thực hiện như sau: Lấy số bị trừ trừ đi số trừ theo từng cột có cùng trọng số với nhau. Trường hợp số bị trừ nhỏ hơn số trừ ta phải vay 1 ở cột cao hơn liền kề của số bị trừ (như vậy sẽ bằng 8 giá trị ở cột cần vay) và tiến hành trừ. Sau đó phải trả lại cột vay bằng cách giảm 1 ở cột cao hơn liền kề của số bị trừ.

* Một số ví dụ:

- Ví dụ 1:	4057	- Ví dụ 2:	7624
	<u>- 034</u>		<u>- 3216</u>
	4023		4406

Ở ví dụ 1, phép trừ diễn ra bình thường, không có cột nào phải vay giá trị của cột cao hơn liền kề. Ở ví dụ 2, ngay cột đầu tiên khi thực hiện phép trừ 4 cho 6 đã không thể thực hiện được. Vì vậy, cần vay ở cột cao hơn liền kề 1 giá trị tương đương 8 đơn vị để trừ cho 6 được kết quả là 2. Kết quả này được cộng với 4 ở cột đầu tiên của số bị trừ, như vậy kết quả ở cột đầu tiên của phép trừ 4 cho 6 là 6. Tiếp theo thực hiện phép trừ ở cột liền kề và số bị trừ ban đầu là 2 lúc này chỉ còn lại giá trị là 1 vì trước đó số bị trừ ở cột thấp liền kề đã vay 1 giá trị.

3. Phép toán cộng, trừ trong hệ đếm mười sáu

Ta đã biết trong hệ đếm mười sáu có 16 chữ số. Ký hiệu 10 chữ số đầu giống như ở hệ mười, sáu chữ số còn lại được ký hiệu là A, B, C, D, E và F. Việc thực hiện các phép toán cộng trừ trong hệ mười sáu diễn ra như trình bày dưới đây:

3.1. Phép toán cộng trong hệ đếm mười sáu

* Để thực hiện cộng hai số ở hệ mười sáu cũng tương tự như ở các hệ đếm mười, hai và tám mà ta đã khảo sát, cụ thể là thực hiện cộng các cột số có cùng trọng số với nhau. Khi kết quả của phép cộng ở cột nào đó mà lớn hơn hoặc bằng 16 ta cần thực hiện các công việc sau:

- Lấy kết quả ở cột đó trừ đi 16.
- Hiệu số của phép trừ trên được ghi vào cột tương ứng của hàng kết quả.
- Cộng 1 vào cột có trọng số cao hơn ngay cạnh cột vừa thực hiện phép cộng có kết quả lớn hơn hoặc bằng mười sáu.

* Một số ví dụ:

- Ví dụ 1:	2C0F	- Ví dụ 2:	3E91
	<u>+ 0226</u>		<u>+ 75D9</u>
	2E35		B46A

Ở ví dụ 1, ngay từ cột đầu tiên (cột có trọng số thấp nhất) khi thực hiện cộng F (có giá trị là 15) với 6 ta được kết quả là 21 (lớn hơn 16), vì vậy ta thực hiện phép trừ: (21 - 16) được kết quả bằng 5, ghi 5 vào cột đầu tiên của hàng

kết quả, đồng thời cộng 1 vào cột có trọng số cao hơn liền kề, dẫn đến ta được kết quả ở cột cao hơn liền kề này là $0 + 2 + 1 = 3$.

3.2. Phép toán trừ trong hệ đếm mười sáu

* Phép toán trừ trong hệ đếm mười sáu được thực hiện như sau: Lấy số bị trừ trừ đi số trừ theo từng cột có cùng trọng số với nhau. Trường hợp số bị trừ nhỏ hơn số trừ ta phải vay 1 ở cột cao hơn liền kề của số bị trừ (như vậy sẽ bằng 16 giá trị ở cột cần vay) và tiến hành trừ. Sau đó phải trả lại cột vay bằng cách giảm 1 ở cột cao hơn liền kề của số bị trừ.

* Một số ví dụ:

$$\begin{array}{r} \text{- Ví dụ 1: } 734F \\ - 559 \\ \hline 6DE6 \end{array}$$

$$\begin{array}{r} \text{- Ví dụ 2: } 66D0 \\ - A04 \\ \hline 5BCC \end{array}$$

IV. CÁC MÃ THÔNG DỤNG

Mã là một quy tắc ký hiệu đặt ra để biểu diễn các thông tin. Mỗi mã gồm một số hữu hạn các từ mã. Mỗi từ mã có một ký hiệu xác định và được gán biểu diễn cho một thông tin.

Trong kỹ thuật số, dạng mã thông dụng là mã số nhị phân. Mỗi từ mã của mã nhị phân là một dãy liên tiếp các số hạng, mỗi số hạng chỉ có thể biểu diễn bằng hai chữ số 0 hoặc 1. Như vậy, một mã nhị phân có độ dài n bit (mỗi từ mã là một dãy gồm n bit) sẽ có 2^n tổ hợp khác nhau và có thể biểu diễn cho 2^n thông tin. Ví dụ mã nhị phân 3 bit có $2^3 = 8$ từ mã là: 000, 001, 010, 011, 100, 101, 110, 111 và có thể biểu diễn cho 8 thông tin.

Việc gán mỗi từ mã biểu diễn cho từng thông tin xác định gọi là mã hoá. Việc làm ngược lại gọi là giải mã.

1. Mã BCD

1.1. Khái niệm về mã BCD

Bảng 1.3. Bảng mã BCD chuẩn

Số TT	Số ở hệ mười	Mã BCD			
		D	C	B	A
1	0	0	0	0	0
2	1	0	0	0	1

3	2	0	0	1	0
4	3	0	0	1	1
5	4	0	1	0	0
6	5	0	1	0	1
7	6	0	1	1	0
8	7	0	1	1	1
9	8	1	0	0	0
10	9	1	0	0	1

Mã BCD là viết tắt của các từ Binary Coded Decimal, đây mã dùng bốn bit (Binary digiT) để mã hoá một chữ số thập phân. Trong hệ đếm mười có mười chữ số, vì vậy trong bộ mã BCD có mười từ mã, mỗi từ mã tương ứng với một chữ số ở hệ mười. Tùy theo trọng số của 4 bit nhị phân được sử dụng, ta có một số loại mã BCD như BCD 8421, BCD 5421, BCD 4221,... Thông dụng nhất là mã BCD 8421 và ta thường quen gọi là mã BCD.

1.2. Bảng mã BCD

Bảng 1.3 là bảng mã BCD chuẩn hay còn gọi là mã BCD 8421, ý nghĩa của bốn chữ số 8, 4, 2 và 1 như sau: bit A có trọng số thấp nhất (bằng 0) nên khi bit A có giá trị bằng 1 thì giá trị của nó ở hệ mười cũng bằng 1 vì $1.2^0 = 1$. Bit B có trọng số bằng 1 nên khi B bằng 1 thì giá trị của nó ở hệ mười là $1.2^1 = 2$. Tương tự như vậy, bit C là 4 và bit D là 8.

Cần lưu ý rằng, mã BCD dùng bốn bit, ký hiệu cho các bit theo trọng số tăng dần là A, B, C và D. Với bốn bit có thể viết được mười sáu tổ hợp khác nhau từ 0000 đến 1111, tuy nhiên chỉ những tổ hợp từ 0000 đến 1001 là thuộc mã BCD còn những tổ hợp còn lại (từ 1010 đến 1111) không thuộc mã BCD.

1.3. Chuyển đổi số ở một số hệ đếm sang mã BCD

1.3.1. Chuyển từ hệ mười sang mã BCD

Mỗi chữ số hệ mười trong số cần chuyển đổi được viết tương ứng với từ mã mà nó có trong bảng mã BCD.

Ví dụ: Số ở hệ mười là 4507 ta sẽ biểu diễn số này ở mã BCD là:

Số ở hệ mười	4	5	0	7
Mã BCD	0100	0101	0000	0111

Ví dụ 2: Số hệ mười là 169, 03 đổi sang mã BCD là:

Số ở hệ mười	1	6	9,	0	3
Mã BCD	0001	0110	1001,	0000	0011

1.3.2. Chuyển đổi mã BCD sang hệ mười

Để chuyển các từ mã BCD sang hệ mười ta làm như sau, nhóm bốn bit liền nhau và đổi thành một chữ số ở hệ mười tương ứng theo bảng mã BCD. Chú ý đối với các bit đứng trước dấu phẩy ta nhóm từ phải sang trái còn các bit sau dấu phẩy thì nhóm từ trái sang phải.

Ví dụ chuyển đổi các mã BCD dưới đây sang hệ đếm mười.

Ví dụ 1: $(1001\ 0011\ 0101)_{\text{BCD}} = (935)_{10}$

Ví dụ 2: $(0111\ 0000\ 0010, 0110\ 1000)_{\text{BCD}} = (702, 68)_{10}$.

1.3.3. Chuyển các số ở hệ hai sang mã BCD và ngược lại

- Chuyển các số ở hệ hai sang mã BCD:

Để chuyển các số ở hệ hai sang mã BCD ta phải thực hiện theo các bước sau:

Bước 1: Chuyển các số ở hệ hai sang hệ mười.

Bước 2: Chuyển các số ở hệ mười sang mã BCD.

Ví dụ cần chuyển số $(101011)_2$ sang mã BCD ta thực hiện như dưới đây.

B1. Chuyển từ hệ hai sang hệ mười:

$$(101011)_2 = (43)_{10}$$

B2. Chuyển từ hệ mười sang mã BCD:

$$(43)_{10} = (0100\ 0011)_{\text{BCD}}$$

- Chuyển mã BCD sang hệ hai:

Để chuyển các mã BCD sang hệ hai, ta cũng thực hiện theo hai bước như sau:

Bước 1: Chuyển mã BCD sang hệ mười.

Bước 2: Chuyển số ở hệ mười sang số ở hệ hai.

Ví dụ: Chuyển mã BCD 0001 0010 0101 sang hệ hai, ta thực hiện:

B1. Chuyển từ mã BCD sang hệ mười:

$$(0001\ 0010\ 0101)_{\text{BCD}} = (125)_{10}$$

B2. Chuyển từ hệ mười sang hệ hai:

$$(125)_{10} = (1111101)_2$$

2. Mã ASCII

2.1. Khái quát chung

Mã ASCII (American National Standard Code for Information Interchange: Mã thông tin chuẩn của quốc gia Mỹ), đây là một mã nhị phân 8 bit thông dụng để mã hoá các ký tự trong xử lý văn bản (các chữ cái, chữ số, các dấu ?, !, >, <, ...). Mã ASCII thường được sử dụng cho thiết bị thông tin và máy tính. Trong tám bit, bit thứ tám là bit parity tức là bit dùng để kiểm tra chẵn lẻ, phát hiện lỗi truyền tin. Số ký tự tối đa có thể mã hoá là $2^7 = 128$.

2.2. Bảng mã ASCII

Dưới đây là bảng mã hoá các ký tự của mã ASCII. Trong bảng chỉ có 95 ký tự, ứng với các từ mã $b_7b_6b_5b_4b_3b_2b_1$ từ 0100001 đến 1111110 là có thể in hoặc hiển thị trên màn hình máy tính. Các từ mã còn lại dùng mã hoá cho các ký tự điều khiển quá trình truyền thông, do đó không in hay hiển thị được.

Ví dụ :

Từ mã	Kí tự điều khiển	Ý nghĩa lệnh điều khiển
0000111	BEL	Chuông
0001001	HT	Lập bảng hàng ngang
0001011	VT	Lập bảng hàng dọc
0001010	LF	Xuống dòng
0100000	SP	Giãn cách
1111111	DEL	Xoá

Bảng 1.4. Mã hoá của mã ASCII

B_4	B_3	B_2	B_1	$B_7B_6B_5$							
				000	001	010	011	100	101	110	111
0	0	0	0	NUL	DLE	SP	0	@	P	\	p
0	0	0	1	SOH	DC1	!	1	A	Q	a	q
0	0	1	0	STX	DC2	"	2	B	R	b	r
0	0	1	1	ETX	DC3	#	3	C	S	c	s
0	1	0	0	EOT	DC4	\$	4	D	T	d	t
0	1	0	1	ENQ	NAK	%	5	E	U	u	u

0	1	1	0	ACK	SYN	&	6	F	V	t	v
0	1	1	1	BEL	ETP	/	7	G	W	g	w
1	0	0	0	BS	CAN	(	8	H	X	h	x
1	0	0	1	HT	EM	)	9	I	Y	i	y
1	0	1	0	LF	SUB	*	:	J	Z	j	z
1	0	1	1	VT	ESC	+	;	K		k	{
1	1	0	0	FF	FS	,	<	L	\	l	
1	1	0	1	CR	GS	-	=	M	]	m	}
1	1	1	0	SO	RS	.	>	N	^	n	~
1	1	1	1	SI	US	`	?	O	-	o	DEL

Bảng dưới đây sẽ giải thích các ký hiệu chữ của mã ASCII

Bảng 1.5. Các ký hiệu chữ của mã ASCII

Kí hiệu	Ý nghĩa	Kí hiệu	Ý nghĩa
NUL	Số không, không, vô hiệu	BS	Lùi một khoảng ký tự
SOH	Bắt đầu của tiêu đề	HT	Kê bảng hướng ngang
STX	Bắt đầu của hành văn	LF	Chuyển dòng
ETX	Kết thúc của hành văn	VT	Kê bảng hướng dọc
EOT	Kết thúc truyền tin	FF	Điều khiển chạy giấy
ENQ	Hỏi	CR	Quay về đầu dòng
ACK	Thừa nhận	SO	Dịch ra (Shift out)
BEL	Chuông	SI	Dịch vào (Shift in)
DLE	Chuyển mã	EM	Hết giấy
DC1	Điều khiển thiết bị 1	SUB	Trừ
DC2	Điều khiển thiết bị 2	ESC	Chuyển mã
DC3	Điều khiển thiết bị 3	FS	Dấu phân cách
DC4	Điều khiển thiết bị 4	GS	Dấu phân cách gói
NAK	Phủ định	RS	Dấu phân cách ghi
SYN	Đồng bộ	US	Dấu phân cách đơn vị
ETB	Kết thúc truyền gói tin	PS	Khoảng trống ký tự
CAN	Huỷ bỏ	DEL	Huỷ bỏ

3. Mã GRAY

3.1. Tổng quan về mã GRAY

Mã GRAY còn gọi là mã vòng. Mã Gray không có trọng số. Ưu điểm chính của mã Gray là chỉ có một bit khác nhau trong hai từ mã kế cận. Nhược điểm của mã này là thiếu trực quan. Mã Gray dùng để biểu diễn các biến trong bảng Karnaugh.

3.2. Chuyển đổi mã

3.2.1. Chuyển từ mã nhị phân sang mã Gray

Khi chuyển mã nhị phân sang mã Gray, ta được mã Gray có số bit bằng đúng số bit của mã nhị phân.

* Phương pháp chuyển đổi:

- *Bước 1:* Hạ chữ số ở cột có trọng số cao nhất của mã nhị phân cần chuyển đổi xuống thành số có trọng số cao nhất cho mã Gray.

- *Bước 2:* Lấy chữ số ở cột có trọng số cao nhất của mã nhị phân cần chuyển đổi cộng với số ở cột liền kề về phía phải, tổng là số tiếp theo của mã Gray. Nếu quá trình cộng có xuất hiện số nhớ thì bỏ số nhớ đi.

- *Bước 3:* Lấy số nhị phân ở cột liền kề phía phải ở bước 2, cộng với số liền kề phía phải tiếp theo và thực hiện như bước 2 cho tới số nhị phân cuối cùng.

Có thể tóm tắt tổng quát quá trình chuyển đổi mã nhị phân sang mã Gray như sau:

Mã nhị phân cần chuyển đổi sang mã Gray có n bit, được ký hiệu: $B_{n-1} B_{n-2} \dots B_1 B_0$. Như vậy, sau khi chuyển đổi xong ta được mã Gray cũng có độ dài n bit, ký hiệu là $G_{n-1} G_{n-2} \dots G_1 G_0$, các bước thực hiện như dưới đây:

$$B1: \quad B_{n-1} = G_{n-1}$$

$$B2: \quad B_{n-1} + B_{n-2} = G_{n-2}$$

$$B3: \quad B_{n-2} + B_{n-3} = G_{n-3}$$

$$\cdot \quad \cdot$$

$$\cdot \quad \cdot$$

$$B_n: \quad B_1 + B_0 = G_0.$$

Ví dụ 1: Chuyển số nhị phân 1011 sang mã Gray:

$$\begin{array}{r}
 101 \\
 + \nearrow \nearrow \nearrow \\
 (1011)_2 \\
 \downarrow \\
 (1110)_G
 \end{array}$$

Ví dụ 2: $(1001)_2 = (?)_G$

$$\begin{array}{r}
 100 \\
 + \nearrow \nearrow \nearrow \\
 (1001)_2 \\
 \downarrow \\
 (1101)_G
 \end{array}$$

Ví dụ 3: $(0101)_2 = (?)_G$

$$\begin{array}{r}
 010 \\
 + \nearrow \nearrow \nearrow \\
 (0101)_2 \\
 \downarrow \\
 (0111)_G
 \end{array}$$

3.2.2. Chuyển từ mã Gray sang số nhị phân

Khi chuyển mã Gray n bit sang số nhị phân, ta được số nhị phân cũng có độ dài n bit.

*Phương pháp chuyển đổi:

- *Bước 1:* Hạ chữ số ở cột số có trọng số cao nhất của mã Gray cần chuyển đổi xuống thành số có trọng số cao nhất cho mã nhị phân.

- *Bước 2:* Lấy chữ số ở cột có trọng số cao nhất của mã nhị phân vừa có đem trừ đi số ở cột tiếp theo của mã Gray liền kề về phía phải với số có trọng số cao nhất của mã Gray, hiệu số là số tiếp theo của mã nhị phân. Nếu quá trình trừ có xuất hiện số nhớ thì bỏ số nhớ đi.

- *Bước 3:* Lấy số nhị phân vừa có tiếp theo, đem trừ đi số tiếp theo của mã Gray ở cột liền kề phía phải và thực hiện như bước 2 cho tới số cuối cùng của mã Gray.

* Có thể tóm tắt tổng quát như sau:

Mã Gray có các bit $G_{n-1}G_{n-2}... G_1G_0$, khi chuyển sang mã nhị phân sẽ có số bit tương ứng $B_{n-1}B_{n-2}... B_1B_0$. Các bước chuyển đổi diễn ra như sau:

$$B1: \quad G_{n-1} = B_{n-1}$$

$$B2: \quad B_{n-1} - G_{n-2} = B_{n-2}$$

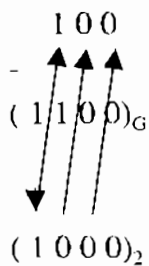
$$B3: \quad B_{n-2} - G_{n-3} = B_{n-3}$$

$$B_n: \quad B_1 - G_0 = B_0$$

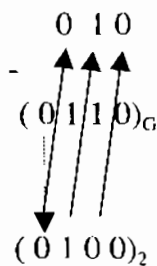
* Ví dụ về việc chuyển mã Gray sang mã nhị phân:

- Ví dụ 1: Chuyển mã Gray 1100 sang mã nhị phân.

Thực hiện như sau:



Ví dụ 2: $(0110)_G = (?)_2$



Bảng 1.6. Bảng mã nhị phân và mã Gray có độ dài bốn bit

Số hệ mười	Mã nhị phân	Mã Gray
0	0000	0000
1	0001	0001
2	0010	0011
3	0011	0010
4	0100	0110
5	0101	0111
6	0110	0101
7	0111	0100
8	1000	1100
9	1001	1101
10	1010	1111
11	1011	1110
12	1100	1010
13	1101	1011
14	1110	1001
15	1111	1000

Câu hỏi

1. Trình bày đặc điểm của các hệ đếm hai, tám, mười và mười sáu.
2. Chuyển các số sau từ hệ mười sang các hệ đếm hai, tám và hệ mười sáu.
a) 81 b) 45 c) 113
d) 38,23 e) 70,54 f) 98,10
3. Chuyển các số hệ hai sau sang các số hệ tám, hệ mười và hệ mười sáu.
a) 10110 b) 11001 c) 10001 d) 11101
e) 1101,001 f) 1100, 110 g) 110011,11
4. Thực hiện các phép toán sau:
* Cộng, trừ ở hệ hai:
a) $1100 + 101$ b) $10101 + 1111$
c) $1001 + 110$ d) $1010 - 111$
* Cộng, trừ ở hệ tám:
a) $7261 + 2215$ b) $4303 + 7033$
c) $4621 - 734$ d) $6742 - 5420$
* Cộng, trừ ở hệ mười sáu:
a) $3566 + C2FF$ b) $230C + E3D$
c) $90A0 - 3FFB$ d) $F032 - 9FFF$
5. Chuyển đổi các số ở hệ đếm mười sang mã BCD:
a) 1047 b) 9532,68 c) 4209,742
6. Chuyển đổi các mã BCD sau sang hệ đếm mười:
a) 100100110100 b) 0101011000000010
c) 011110000011, 00100101 d) 00111001, 100000000110
7. Chuyển đổi mã nhị phân sau sang mã Gray:
a) 1010 b) 1100 c) 1001
8. Chuyển đổi mã Gray sang mã nhị phân:
a) 1011 b) 1101 c) 1000

Chương 2

ĐẠI SỐ LOGIC

Mục tiêu

- Học sinh hiểu được các khái niệm về biến và hàm logic, thực hiện được các phép toán cơ bản trong đại số logic. Hiểu được nguyên lý làm việc của các cổng logic, biết biểu diễn, tối thiểu hàm logic và có khả năng phân tích, thiết kế được các mạch tổ hợp.

- Rèn luyện tính kiên trì, cẩn thận và tư duy logic.

Nội dung trọng tâm

- Các phép toán trong đại số logic.
- Ký hiệu và nguyên lý làm việc của các cổng logic.
- Các phương pháp biểu diễn hàm logic.
- Các phương pháp tối thiểu hàm logic.
- Phân tích và thiết kế mạch tổ hợp.

I. CÁC KHÁI NIỆM VÀ PHÉP TOÁN LOGIC

1. Khái niệm về biến và hàm logic

Một khái niệm làm nền tảng của đại số logic là khái niệm về các mệnh đề. *Biến logic* là một khái niệm dùng thay cho thuật ngữ *mệnh đề tùy ý*, các mệnh đề này có thể đúng (Đ) hoặc sai (S) và không có khả năng một mệnh đề vừa đúng lại vừa sai, do đó biến logic chỉ có thể nhận một trong hai giá trị Đ hoặc S. Người ta thường ký hiệu các biến logic là A, B, C,... hoặc $x_1, x_2, x_3...$

Ví dụ:

A = Nguyễn Văn X là học sinh giỏi

B = Nguyễn Thị Y là học sinh giỏi

Từ hai biến logic trên, ta nhận thấy rằng:

- Nếu X là học sinh giỏi thì A = Đ
- Nếu X không phải là học sinh giỏi thì A = S

- Nếu Y là học sinh giỏi thì $B = Đ$
- Nếu Y không phải là học sinh giỏi thì $B = S$.

Nếu kết hợp với phép toán nhân logic thì biểu thức $F = A.B$ sẽ nhận giá trị Đ khi cả X lẫn Y đều là học sinh giỏi, các trường hợp khác thì F đều nhận giá trị là S. Đây chính là một ví dụ đơn giản về một hàm có hai biến logic $F(A,B)$.

Trong đó $F(A,B) = A.B$, (phép nhân logic).

Một cách tổng quát, một mệnh đề phức tạp được tạo thành từ những mệnh đề đơn giản ban đầu, chúng sẽ chỉ có thể nhận một trong hai giá trị là Đ hoặc S; trong trường hợp này ta có thể biểu diễn dưới dạng $F(A,B,C,...)$ hay $F(x_1,x_2,x_3,...)$ và ta gọi chúng là hàm logic của các biến A,B,C,... hay của $x_1,x_2,x_3,...$.

- Các ký hiệu 0,1:

Trong kỹ thuật số, các giá trị Đ (đúng) và S (sai) của một biến logic thường được ký hiệu là 1 và 0. Cần lưu ý rằng 1 và 0 ở đây là các ký hiệu mà không phải là các chữ số hệ hai. Việc sử dụng các ký hiệu này, như ở những chương sau sẽ thấy, tỏ ra thuận tiện và đơn giản hơn so với việc sử dụng Đ và S.

Biện pháp kỹ thuật để thực hiện các giá trị logic (mà trong trường hợp đang xét là giá trị 1 và 0) phụ thuộc vào việc chọn các giá trị số vật lý để biểu diễn. Thông thường người ta chọn tín hiệu điện thế để biểu diễn, chẳng hạn với các vi mạch thuộc họ TTL thì mức +5V biểu diễn 1 logic còn mức 0V biểu diễn 0 logic, nhưng với các vi mạch CMOS thì 1 logic thể hiện mức từ +3V đến +18V và 0 logic thể hiện mức 0V đối với logic dương (xem ở mục II. Các cổng logic cơ bản). Vì vậy, từ đây về sau ta sẽ sử dụng các ký hiệu 0 và 1 để thể hiện giá trị của biến và hàm logic.

Tóm lại, trong đại số logic, biến và hàm đều chỉ nhận 2 giá trị là 0 và 1.

2. Các hàm logic sơ cấp (hàm logic cơ bản)

2.1. Hàm logic sơ cấp một biến

Gọi $F(A)$ là hàm logic của một biến A. Vì A có thể nhận một trong hai giá trị 1 hoặc 0 cho nên có thể xuất hiện bốn trường hợp như sau đối với hàm $F(A)$ (xem bảng 2.1).

Bảng 2.1: Các trường hợp cho hàm logic 1 biến

F_i	A		F(A)	
	0	1	Biểu thức	Tên gọi

F_1	0	0	0	Hằng số
F_2	0	1	A	Lặp lại A
F_3	1	0	$\bar{A}$	Đảo biến A
F_4	1	1	1	Hằng số

Trong các hàm nêu trên, ta lưu ý đến hai hàm sau:

+ Hàm $F_2(A) = A$ lặp lại biến A.

+ Hàm đảo $F_3(A) = \bar{A}$ nhận giá trị là đảo của biến A.

2.2. Hàm logic hai biến

Vì các biến A, B có thể nhận một trong hai giá trị logic là 1 hoặc 0 cho nên sẽ tồn tại $2^2 = 4$ bộ giá trị của A, B và như vậy sẽ có $2^4 = 16$ hàm logic khác nhau mà ta ký hiệu từ F_0 đến F_{15} như trên bảng 2.2.

Bảng 2.2: Các trường hợp cho hàm logic 2 biến

A	0	0	1	1	Ký hiệu và biểu thức đại số của hàm số	Tên gọi của hàm số
B	0	1	0	1		
F_0	0	0	0	0	$F_0 = 0$	Hằng số 0
F_1	0	0	0	1	$F_1 = AB$	Nhân logic
F_2	0	0	1	0	$F_2 = A\bar{B}$	Cấm B
F_3	0	0	1	1	$F_3 = A$	Bảng A
F_4	0	1	0	0	$F_4 = \bar{A}B$	Cấm A
F_5	0	1	0	1	$F_5 = B$	Bảng B
F_6	0	1	1	0	$F_6 = A \oplus B = A\bar{B} + \bar{A}B$	Khác dấu
F_7	0	1	1	1	$F_7 = A+B$	Cộng logic
F_8	1	0	0	0	$F_8 = A \downarrow B = \overline{A+B}$	Hàm Pierce
F_9	1	0	0	1	$F_9 = A \sim B = AB + \bar{A}\bar{B}$	Đồng dấu
F_{10}	1	0	1	0	$F_{10} = \bar{B}$	Bù của B
F_{11}	1	0	1	1	$F_{11} = B \rightarrow A = A + \bar{B}$	Kéo theo A
F_{12}	1	1	0	0	$F_{12} = \bar{A}$	Bù của A
F_{13}	1	1	0	1	$F_{13} = A \rightarrow B = \bar{A} + B$	Kéo theo B
F_{14}	1	1	1	0	$F_{14} = A/B = \bar{A}B$	Hàm Sheffer
F_{15}	1	1	1	1	$F_{15} = 1$	Hằng số 1

Trong hệ hàm logic hai biến này, ta lưu ý đến các hàm sau:

+ **Hàm $F_1(A,B) = AB$**

Hàm này thực hiện phép toán nhân logic của hai biến A và B.

Trong kỹ thuật, các phân tử mạch để thực hiện chức năng của hàm F_1 được gọi là phân tử AND (phân tử “Và”).

+ **Hàm $F_6 = A \oplus B = \overline{A}B + A\overline{B}$**

Hàm F_6 được coi như tổng logic của hai hàm F_2 và F_4 ở bảng 2.2. Ta có thể xây dựng mạch thực hiện chức năng hàm F_6 từ hai phân tử cấm (hàm F_2 và F_4) và phân tử OR (hàm F_7). Hàm còn có tên gọi là hàm XOR-(*Exclusiv OR*).

+ **Hàm $F_7 = A + B$**

Thực hiện phép cộng logic đối với hai biến A và B. Phân tử mạch dùng để thực hiện hàm này gọi là phân tử OR (phân tử “hoặc”).

+ $F_8 = A \downarrow B = \overline{A+B}$

Tên gọi là hàm hoặc - đảo (NOR)

+ **Hàm $F_{14} = A/B = \overline{AB}$**

Còn được gọi là hàm Sheffer. (“Và - đảo” - NAND)

3. Các công thức và định lý cơ bản trong đại số logic

3.1. Công thức về quan hệ giữa các hằng số

Vì trong đại số logic chỉ có hai hằng số 0,1 các biến logic cũng chỉ lấy một trong hai giá trị đó, và cũng chỉ có ba phép toán logic cơ bản nhất, cho nên cũng chỉ có các quan hệ dưới đây:

Công thức 1	$0 \cdot 0 = 0$
Công thức 1'	$1 + 1 = 1$
Công thức 2	$0 \cdot 1 = 0$
Công thức 2'	$1 + 0 = 1$
Công thức 3	$1 \cdot 1 = 1$
Công thức 3'	$0 + 0 = 0$
Công thức 4	$\overline{0} = 1$
Công thức 4'	$\overline{1} = 0$

Những quan hệ trên đây giữa hai hằng số làm tiền đề của đại số logic. Nghĩa là, chúng là các quy tắc phép toán cơ bản đối với tư duy logic.

3.2. Quan hệ giữa biến số và hằng số

Công thức 5	$A \cdot 1 = A$
Công thức 5'	$A + 0 = A$
Công thức 6	$A \cdot 0 = 0$
Công thức 6'	$A + 1 = 1$
Công thức 7	$A \cdot \bar{A} = 0$
Công thức 7'	$A + \bar{A} = 1$

3.3. Các định lý tương tự đại số thường

Luật giao hoán

Công thức 8	$A \cdot B = B \cdot A$
Công thức 8'	$A + B = B + A$

Luật kết hợp

Công thức 9	$(A \cdot B) \cdot C = A \cdot (B \cdot C)$
Công thức 9'	$(A + B) + C = A + (B + C)$

Luật phân phối

Công thức 10	$A \cdot (B + C) = A \cdot B + A \cdot C$
Công thức 10'	$A + BC = (A + B) \cdot (A + C)$

3.4. Các định lý đặc thù chỉ có trong đại số logic

Luật đồng nhất

Công thức 11	$A \cdot A = A$
Công thức 11'	$A + A = A$

Định lý De Morgan

Công thức 12	$\overline{A \cdot B} = \bar{A} + \bar{B}$
Công thức 12'	$\overline{A + B} = \bar{A} \cdot \bar{B}$

Luật hoàn nguyên

Công thức 13	$\overline{\overline{A}} = A$
--------------	-------------------------------

3.5. Một số công thức thường dùng

Công thức 14	$A + \bar{A} \cdot B = A + B$
Công thức 15	$A \cdot B + \bar{A} C + BC = AB + \bar{A} C$

Phương pháp chứng minh các công thức trên là lập bảng tất cả các giá trị có thể của các biến và tính tương ứng với vế phải, vế trái riêng rẽ. Nếu đẳng thức giữa hai vế tồn tại với tất cả các giá trị có thể thì công thức là đúng. Công thức 5 và công thức 13 rất dễ chứng minh. Dưới đây sẽ chứng minh làm mẫu các công thức 10' và công thức 12.

Ví dụ 1. Chứng minh công thức 10':

$$A + B . C = (A + B) . (A + C)$$

Bảng 2.3. Bảng chứng minh công thức 10'

A	B	C	B.C	A+B.C	A+B	A+C	(A+B).(A+C)
0	0	0	0	0	0	0	0
0	0	1	0	0	0	1	0
0	1	0	0	0	1	0	0
0	1	1	1	1	1	1	1
1	0	0	0	1	1	1	1
1	0	1	0	1	1	1	1
1	1	0	0	1	1	1	1
1	1	1	1	1	1	1	1

Giải: Lập bảng (bảng 2.3) tất cả các giá trị có thể có của biến và tính như sau:

Tất cả các giá trị của 3 biến A, B, C tạo thành 8 tổ hợp. Bảng chân lý của hàm $A + B.C$ trùng với bảng chân lý của hàm $(A+B).(A+C)$. Vậy công thức $A + B.C = (A+B).(A+C)$ đã được chứng minh.

Ví dụ 2: Chứng minh công thức 12:

$$\overline{A.B} = \overline{A} + \overline{B}$$

Giải: Lập bảng (bảng 2.4) tất cả các giá trị có thể có của biến và tính như sau:

Bảng 2.4. Bảng chứng minh công thức 12

A	B	A.B	$\overline{A.B}$	$\overline{A}$	$\overline{B}$	$\overline{A} + \overline{B}$
0	0	0	1	1	1	1
0	1	0	1	1	0	1
1	0	0	1	0	1	1
1	1	1	0	0	0	0

Tất cả các giá trị của 2 biến A, B tạo thành 4 tổ hợp. Bảng chân lí của hàm $\overline{A.B}$ trùng với bảng chân lí của hàm $\overline{A} + \overline{B}$. Vậy công thức 12 đã được chứng minh.

II. CÁC CỔNG LOGIC CƠ BẢN

1. Các khái niệm

1.1. Định nghĩa

Các cổng logic cơ bản (các phần tử logic cơ bản) là các mạch điện tử dùng để thực hiện chức năng của các hàm logic cơ bản. Gọi là cổng vì người ta hay dùng các phần tử logic làm cửa ngõ cho thông tin đi qua.

Các cổng logic cơ bản là các phần tử đóng vai trò chủ yếu để thực hiện các chức năng logic đơn giản nhất trong các sơ đồ logic, là các sơ đồ thực hiện một hàm logic nào đó.

Từ các cổng logic cơ bản ta có thể kết hợp lại để tạo ra nhiều mạch logic thực hiện các hàm logic phức tạp hơn. Các biến A, B, ..., và hàm F chỉ nhận các giá trị logic là *Đúng* (hay 1) và *Sai* (hay 0).

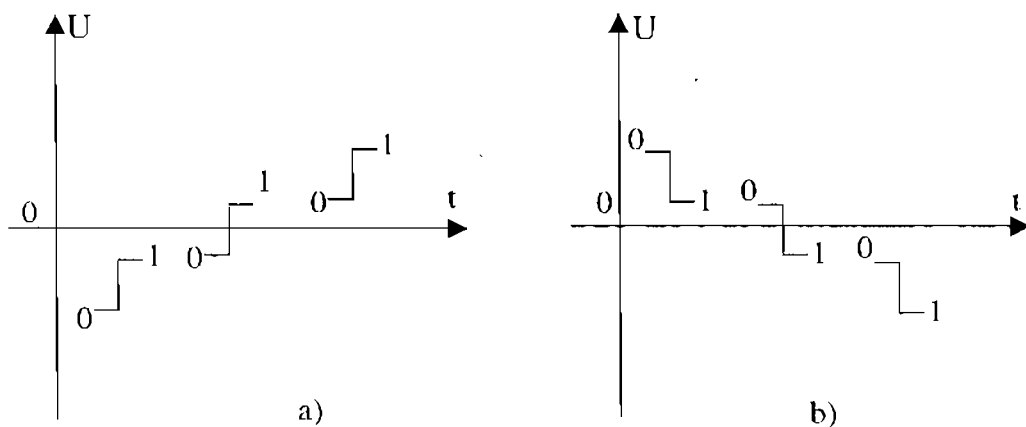
1.2. Các đại lượng vật lý để mô tả tín hiệu vào ra

Người ta thường dùng các tín hiệu điện để biểu diễn tín hiệu vào ra của các cổng logic nói riêng và của các mạch logic nói chung. Chúng có thể là "tín hiệu điện thế" hoặc "tín hiệu xung".

* *Biểu diễn bằng tín hiệu điện thế*: Dùng hai mức điện thế khác nhau để biểu diễn hai giá trị *Đúng* (1) hoặc *Sai* (0) như trên hình 2.1. Có hai phương pháp biểu diễn hai giá trị này:

- + Phương pháp logic dương (Hình 2.1.a):
 - Điện thế *dương hơn* hay *ít âm hơn* là *Đúng* (1).
 - Điện thế *ít dương hơn* hay *âm hơn* là *Sai* (0).
- + Phương pháp logic âm (Hình 2.1.b):
 - Điện thế *âm hơn* hay *ít dương hơn* là *Đúng* (1).
 - Điện thế *ít âm hơn* hay *dương hơn* là *Sai* (0).

Phương pháp logic thông dụng hơn và sẽ được sử dụng trong toàn bộ cuốn sách này, trừ một vài trường hợp đặc biệt sẽ có ghi chú.



Hình 2.1: Biểu diễn các mức logic bằng tín hiệu điện thế

* *Biểu diễn bằng tín hiệu xung*: Hai giá trị logic Đúng (1) hoặc Sai (0) tương ứng với sự xuất hiện hoặc không xuất hiện của xung trong dãy tín hiệu theo một chu kỳ T_x nhất định (Hình 2.2). Để đảm bảo sự ổn định của chu kỳ T_x này người ta đưa vào một trong các đầu vào của mạch logic một dãy xung có độ ổn định rất cao gọi là *xung nhịp* (hay xung đồng hồ: clock pulse).

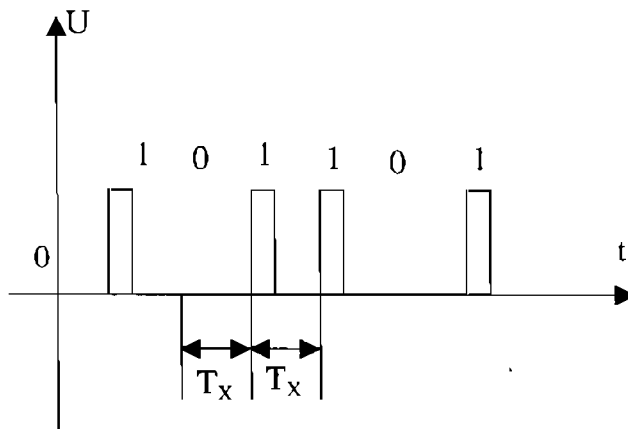
Trong các mạch logic sử dụng dữ liệu là tín hiệu xung, các xung thường có độ rộng sườn và biên độ ở trong một giới hạn cho phép nào đó phù hợp với từng trường hợp cụ thể phụ thuộc vào phương pháp biểu diễn mã và loại phần tử logic được sử dụng cũng như độ phức tạp của các sơ đồ mạch.

1.3. Bảng mức độ điện thế

Khi khảo sát các mạch điện tử logic, để được tổng quát, người ta thường dùng một bảng chỉ mức độ điện thế CAO (ký hiệu là H: High) hay THẤP (ký hiệu là L: Low) của các dữ liệu. Sự cao thấp này được đánh giá theo chiều từ dưới lên trong ý niệm thông thường về điện thế. Đôi khi, nếu không sợ nhầm lẫn, ta cũng có thể gọi các bảng mức độ điện thế là bảng chân lý.

Ví dụ:

Một cổng logic có bảng mức độ điện thế như trên hình 2.3.a sẽ tương ứng với hai phương pháp biểu diễn bằng tín hiệu điện thế: dùng logic dương (Hình 2.3.b) và dùng logic âm (Hình 2.3.c) trong đó A,B là dữ liệu vào và F là dữ liệu ra.



Hình 2.2: Biểu diễn các mức logic bằng tín hiệu xung

A B	F
L L	H
L H	H
H L	H
H H	L

a)

A B	F
0 0	1
0 1	1
1 0	1
1 1	0

b)

A B	F
1 1	0
1 0	0
0 1	0
0 0	1

c)

Hình 2.3: Ví dụ dùng bảng chỉ mức độ điện thế

1.4. Công nghệ chế tạo các cổng logic

Hiện nay về công nghệ chế tạo, người ta thường chia ra thành hai loại sơ đồ là sơ đồ dùng linh kiện rời và sơ đồ dùng mạch tích hợp (Integrated Circuits: IC). Tuy nhiên, vì sơ đồ dùng linh kiện rời ngày càng ít được sử dụng, cho nên ở các phần dưới đây chỉ giới thiệu các loại mạch logic được chế tạo ở dạng mạch tích hợp.

Bảng 2.5. Bảng tóm tắt các đặc tính của các cổng logic dạng tích hợp

Đặc tính	RTL	DTL	HTL	TTL	ECL	RMOS	CMOS
Cổng cơ bản	NOR	NAND	NAND	NAND	OR-NOR	NAND	NOR-NAND
Trở kháng ra khi đầu ra Z cao (Ω)	640	6k/2k	15k/1,5k	70	15	2k	1,5k
Trở kháng ra khi đầu ra Z thấp (Ω)	R_{bh}	R_{bh}	R_{bh}	R_{bh}	15	25k	1,5k
Hệ số mắc tải	5	8	10	10	25	-	>50
Nguồn cung cấp (V)	3,6 $\pm$ 10%	5 $\pm$ 10%	15 $\pm$ 1V	5 $\pm$ 10%	-5,2 $\pm$ 10%	-27 $\pm$ 10%	+3+18V
Công suất tiêu thụ đối với một cổng (mW)	12	8	22	10	40	0,1	0,1 đến 1
Mức độ chống tạp âm	Thường	Tốt	Tốt	Tốt	Tốt	Khá	Rất tốt
Thời gian chậm trễ đối với một cổng (ns)	12	30	90	10	2	700	30
Tần số xung nhịp cực đại (MHz)	8	12	4	35	60	1	10

2. Các đặc tính tiêu biểu của cổng logic

2.1. Bảng tóm tắt đặc tính

Bảng 2.5 tóm tắt những đặc tính tiêu biểu của các cổng logic được chế tạo theo phương pháp mạch tích hợp.

2.2. Giải thích đặc tính

2.2.1. Cổng cơ bản (Basic gate)

Loại cổng logic thông dụng nhất tương ứng với phương pháp chế tạo đó. Dĩ nhiên có thể chế tạo các mạch phức tạp, nhưng thường các mạch này là do sự kết hợp của các cổng logic cơ bản.

2.2.2. Trở kháng ra (Output impedance)

Trở kháng ra thay đổi theo trạng thái cao hay thấp của đầu ra Z và tùy ở dạng mạch đầu ra. Nếu mạch dùng một transistor thì khi đầu ra Z cao trở kháng ra coi như bằng điện trở cực góp (và các điện dung ký sinh liên hệ), khi Z thấp thường là điện trở bão hoà (R_{bh}) của transistor (chẳng hạn loại TTL) thì trở kháng ra thấp cho cả hai trạng thái của Z .

2.2.3. Hệ số mắc tải (Fan out)

Là số cổng logic có thể nối được vào đầu ra của một cổng logic khác. Hệ số mắc tải thường chỉ được xác định cho các cổng logic cùng loại, chẳng hạn hệ số mắc tải của DTL là số NAND DTL có thể nối vào đầu ra của một NAND DTL. Hệ số mắc tải là một số xác định phụ thuộc vào đặc tính đầu ra và của cổng logic.

2.2.4. Nguồn cung cấp (Power supply)

Nguồn cung cấp cần được chọn sao cho thoả mãn các điều kiện sử dụng cổng logic. Đối với loại mạch tích hợp, nguồn cung cấp thường được lựa chọn phụ thuộc vào phương pháp chế tạo và theo các điều kiện tối ưu về mức độ chống tạp âm, về công suất tiêu thụ, về tốc độ chuyển đổi trạng thái, v.v... Thông thường, khi các đầu vào và đầu ra của cổng logic thay đổi đột ngột có thể tạo ra những rối loạn cho các mạch khác, vì thế cần phải lọc nguồn cung cấp bằng các tụ điện vào khoảng từ $0,1\mu\text{F}$ đến $1\mu\text{F}$.

2.2.5. Công suất tiêu thụ đối với một cổng (Power dissipated per gate)

Là công suất tiêu thụ trong toàn mạch tích hợp. Trị số này càng lớn khi mạch càng có nhiều điện trở và nhiều transistor làm việc ở chế độ bão hoà. Loại PMOS và CMOS có điện trở trong lớn và thường làm việc theo kiểu "đẩy-kéo" (*push-pull*) nên ít tiêu thụ công suất nhất. Mặc dù mỗi cổng chỉ tiêu thụ công suất ở mức miliWatt (mW) nhưng con số này lại trở nên đáng kể bởi vì các thiết bị điện tử số có thể phải dùng đến hàng ngàn cổng, và như vậy công suất tiêu thụ tổng cộng có thể lớn. Công suất tiêu thụ là một căn cứ để thiết kế nguồn cung cấp và tản nhiệt cho mạch. Cũng cần lưu ý thêm rằng, công suất

tiêu thụ thay đổi theo tần số tín hiệu bởi vì cổng logic là mạch sử dụng các tín hiệu chuyển đổi trạng thái.

2.2.6. Mức độ chống tạp âm (Noise immunity level)

Là biên độ tạp âm tối đa có thể vào mạch mà không làm đầu ra thay đổi trạng thái. Trị số này thường được tính theo tiêu chuẩn sau:

- Giữa các cổng logic cùng loại.
- Trạng thái hoạt động kém nhất (về tiêu chuẩn chế tạo, về điện thế, số tải mắc vào cực đại, v.v...)

Tiêu chuẩn thứ hai này khẳng định rằng bình thường thì cổng logic có mức độ chống tạp âm cao hơn so với mức độ tiêu chuẩn.

Vì đặc tính của các cổng logic thay đổi theo nhiệt độ nên mức độ chống tạp âm cũng thay đổi theo. Khi nhiệt độ tăng mức độ này giảm.

2.2.7. Thời gian chậm trễ đối với một cổng (Propagation delay per.gate)

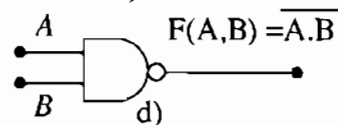
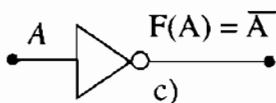
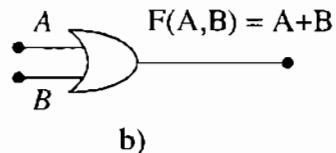
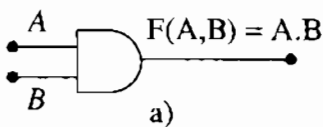
Là khoảng thời gian từ lúc mạch nhận được tín hiệu tại đầu vào cho đến lúc đầu ra thay đổi trạng thái. Thời gian này đối với một cổng logic thường nhỏ nhưng sau khi qua nhiều mạch thì trở nên đáng kể và trở thành yếu tố quyết định tần số hoạt động tối đa của thiết bị.

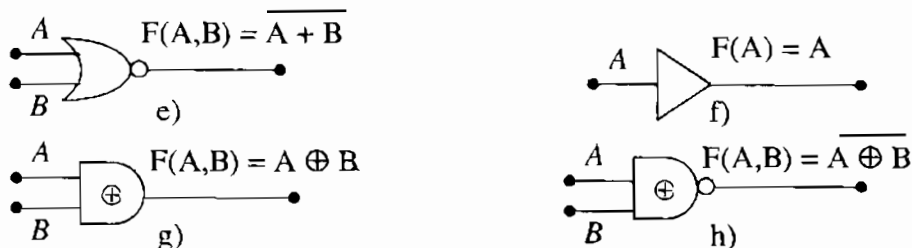
Đôi khi nhà chế tạo cho biết thời gian chậm trễ qua một cặp cổng (pair delay), tức là trị số thời gian chậm trễ trung bình sau khi truyền qua hai cổng liên tiếp. Cách này có ưu điểm vì chia đều sự chậm trễ của hai trạng thái lên cao và xuống thấp.

2.2.8. Tần số xung nhịp cực đại (Maximum clock rate)

Các cổng logic có thể được mắc để tạo thành mạch tạo xung nhịp (flip-flop). Tùy theo tốc độ chuyển mức logic nhanh hay chậm mà có thể tạo thành các mạch có tần số làm việc cao hay thấp.

3. Các cổng logic cơ bản





Hình 2.4: Các cổng logic cơ bản

3.1. Ba phân tử logic cơ bản

Ba phân tử logic cơ bản là các mạch logic đơn giản nhất để thực hiện 3 phép tính cơ sở trên các biến logic: phép nhân logic (AND), phép cộng logic (OR) và phép tính phủ định (NOT).

3.1.1. Phần tử nhân logic (“và”, AND)

Là mạch điện thực hiện phép tính nhân logic giữa 2 biến A và B ở đầu vào. Đầu ra nhận được giá trị bằng $A.B$.

Sơ đồ ký hiệu phần tử nhân logic hình 2.4.a. Bảng 2.6 là bảng chân lý của phần tử nhân logic. Qua bảng chân lý chúng ta thấy đầu ra của phần tử AND chỉ bằng 1 logic khi tất cả các đầu vào bằng 1 logic.

Bảng 2.6

Đầu vào		Đầu ra
A	B	$A.B$
0	0	0
0	1	0
1	0	0
1	1	1

Suy rộng ra, nếu một phần tử AND có n đầu vào, đầu ra của nó chỉ bằng 1 logic khi tất cả n đầu vào đều bằng 1 logic.

3.1.2. Phần tử cộng logic (“hoặc”, OR)

Bảng 2.7

Là mạch điện thực hiện phép tính cộng logic giữa 2 biến A và B ở đầu vào. Đầu ra nhận được giá trị bằng $A + B$.

Sơ đồ ký hiệu phần tử cộng logic hình 2.4.b. Bảng 2.7 là bảng chân lý của phần tử cộng logic.

Bảng chân lý cho thấy đầu ra của phần tử OR chỉ bằng 0 logic khi tất cả các đầu vào bằng 0 logic.

Đầu vào		Đầu ra
A	B	$A+B$
0	0	0
0	1	1
1	0	1
1	1	1

Suy rộng ra, nếu một phần tử OR có n đầu vào, đầu ra bằng 0 logic khi tất cả n đầu vào đều bằng 0 logic.

3.1.3. Phần tử đảo logic (“phủ định”, NOT)

Là mạch điện thực hiện phép tính đảo biến logic A ở đầu vào. Đầu ra nhận được giá trị bằng A đảo.

Sơ đồ ký hiệu phần tử đảo logic hình 2.4.c. Bảng 2.8 là bảng chân lý của phần tử đảo logic.

Bảng 2.8

Đầu vào	Đầu ra
A	A
0	1
1	0

3.2. Các phần tử logic vạn năng

3.2.1. Phần tử “và - đảo” (NAND)

Là mạch điện thực hiện phép tính và - đảo logic giữa 2 biến A và B ở đầu vào. Đầu ra nhận được giá trị bằng $\overline{AB}$.

Sơ đồ ký hiệu phần tử và - đảo logic hình 2.4.d. Bảng 2.9 là bảng chân lý của phần tử và - đảo.

Bảng 2.9

Đầu vào		Đầu ra
A	B	$\overline{AB}$
0	0	1
0	1	1
1	0	1
1	1	0

Qua bảng chân lý chúng ta thấy phần tử NAND chỉ bằng 0 logic khi tất cả các đầu vào bằng 1 logic. Suy rộng ra, nếu một phần tử NAND có n đầu vào, đầu ra bằng 0 logic khi tất cả n đầu vào đều bằng 1 logic.

3.2.2. Phần tử “hoặc - đảo” (NOR)

Là mạch điện thực hiện phép tính hoặc - đảo logic giữa 2 biến A và B ở đầu vào. Đầu ra nhận được giá trị $= \overline{A+B}$.

Sơ đồ ký hiệu phần tử hoặc - đảo logic hình 2.4.e. Bảng 2.10 là bảng chân lý của phần tử hoặc - đảo.

Bảng 2.10

Đầu vào		Đầu ra
A	B	$\overline{A+B}$
0	0	1
0	1	0
1	0	0
1	1	0

Bảng chân lý cho thấy phần tử NOR chỉ bằng 1 logic khi tất cả các đầu vào bằng 0 logic.

Suy rộng ra, nếu một phần tử NOR có n đầu vào, đầu ra bằng 1 logic khi tất cả n đầu vào đều bằng 0 logic.

3.3. Phần tử tương đương và không tương đương

3.3.1 Phần tử không tương đương

Phần tử không tương đương hay còn gọi là phần tử XOR hai đầu vào (viết tắt của từ EXCLUSIVE OR) là mạch logic có hai đầu vào và một đầu ra, dùng để thực hiện phép tính HOẶC - LOẠI TRỪ trên hai biến logic A và B:

$$F = A \oplus B$$

Sơ đồ ký hiệu phần tử XOR hình 2.4.g.

Bảng chân lý của phần tử này là bảng 2.11.

3.3.2. Phần tử tương đương

Phần tử tương đương hay còn gọi là phần tử XOR đảo, là mạch logic có hai đầu vào và một đầu ra, dùng để thực hiện hai phép tính liên tiếp: thứ nhất là phép tính XOR, thứ hai phép tính đảo. Đầu ra có giá trị:

$$F = \overline{A \oplus B}$$

Hình 2.4.h là sơ đồ ký hiệu phần tử tương đương.

Bảng 2.11

Đầu vào		Đầu ra
A	B	$A \oplus B$
0	0	0
0	1	1
1	0	1
1	1	0

Bảng 2.12

Đầu vào		Đầu ra
A	B	$\overline{A \oplus B}$
0	0	1
0	1	0
1	0	0
1	1	1

Bảng chân lý của phần tử này là bảng 2.12

- Hình 2.4.f là cổng đệm (Buffer), hàm chức năng nó thực hiện là $F(A) = A$. Nó có tác dụng dùng để sửa dạng xung và khuếch đại tín hiệu.

4. Mạch ba trạng thái

Các mạch ra của vi mạch ngày càng được cải tiến để đáp ứng nhu cầu sử dụng ngày càng cao của thực tế. Trước khi tìm hiểu về mạch ra ba trạng thái ta xét sơ bộ một số loại mạch ra khác cũng của vi mạch loại TTL - là loại được sử dụng rộng rãi.

* *Tăng ra bình thường (kéo lên thụ động)*

Mạch có sơ đồ như hình 2.5, trong đó C_1 là tụ ký sinh gồm $C_{lắp ráp}$ và $C_{tải}$.

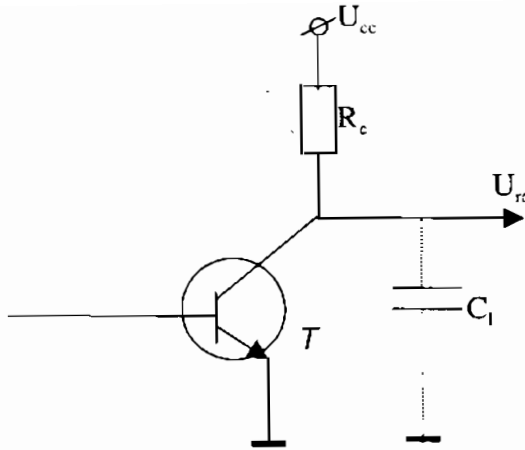
Nguyên lý:

- Khi T tắt, $U_{ra} = "H" \sim 5V$.

Tụ C_1 được nạp, thời gian nạp $T_n = C_1 \cdot R_c$.

- Khi T thông bão hoà, $U_{ra} = U_{cebh} \sim 0V = "L"$.

Tụ C_1 phóng điện, $T_f = C_1 \cdot R_{cebh}$.



Hình 2.5: Mạch ra kéo lên thụ động

- Nhận thấy rằng khi T tắt, T_n lớn, thời gian quá độ tăng, giảm tốc độ chuyển biến của mạch. Nhưng nếu giảm R_c , T làm việc ở chế độ tích cực, công suất tiêu tán trên mạch tăng. Do vậy mạch này ít được dùng, phải cải tiến mạch ra.

* Mạch ra cột chạm hay kéo lên tích cực

Mạch có sơ đồ như hình 2.6

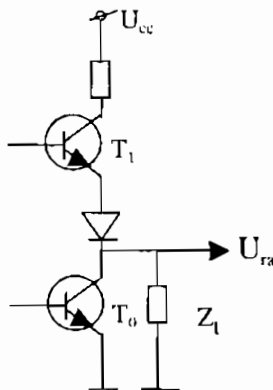
Mạch có đặc điểm $Z_{ra} = Z_{T \text{ bão hoà}}$, do vậy Z_{ra} luôn luôn nhỏ, cụ thể:

$Z_{ra1} = Z_{T1 \text{ bão hoà}}$; $U_{ra} = \text{"H"}$

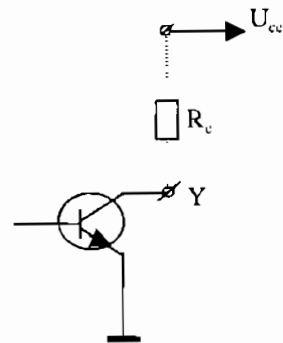
$Z_{ra1} = Z_{T0 \text{ bão hoà}}$; $U_{ra} = \text{"L"}$

Ưu điểm: Tốc độ chuyển mạch cao.

Nhược điểm: Có khả năng hỏng mạch khi nối chung các đầu ra.



Hình 2.6: Mạch ra kéo lên tích cực



Hình 2.7: Mạch ra hở collector

** Mạch ra hở collector*

Sơ đồ nguyên lý cho ở hình 2.7. Mạch có cực góp để hở có thể sử dụng như mạch logic thông thường bằng cách mắc thêm R_c lên nguồn. Giá trị R_c phụ thuộc vào tải mắc ở đầu ra. Tải có thể là mạch TTL khác hay đèn LED... Thông thường giá trị R_c bằng vài trăm Ω đến vài kilo Ω .

- Ưu điểm:

+ Có thể chọn R_c thích hợp với tình trạng của tải.

+ Có thể nối chung các đầu ra và dùng một điện trở R_c . Lúc này điểm nối chung thực hiện chức năng AND với mức “H” các đầu ra.

** Mạch ra ba trạng thái*

Các mạch ra trên đều có nhược điểm: Luôn luôn chịu nhiều ảnh hưởng của tải. Các mạch ra kéo lên tích cực hay hở collector có trở kháng ra khi T tắt lớn, do vậy thời gian quá độ của mạch lớn. Mạch ra cột chạm tuy có trở kháng ra tại các trạng thái 0 và 1 luôn luôn nhỏ nhưng lại chịu nhiều ảnh hưởng của tải, mặt khác không cho phép nối chung các đầu ra.

Năm 1969 tại trường đại học Toyota, người ta đã phát minh ra mạch ra ba trạng thái. Ba trạng thái đó là:

- Trạng thái 0

- Trạng thái 1

Hai trạng thái này như là mạch chạm cột. Trạng thái thứ ba:

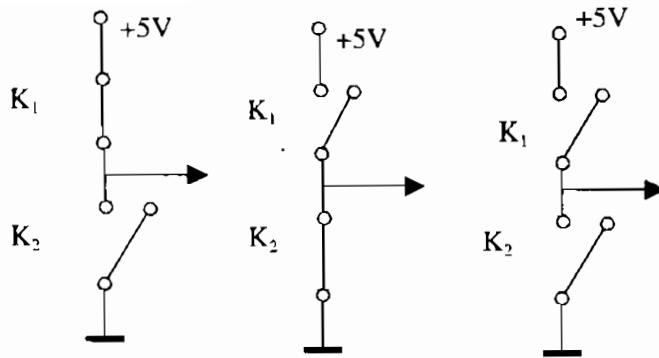
Trạng thái treo: trở kháng cao. Sơ đồ tầng ra cuối cùng hoàn toàn giống như tầng ra cột chạm, trong đó hai transistor T_1 và T_2 tương đương hai khoá điện tử (hình 2.8). Chỉ khác là nhờ tín hiệu điều khiển có thể tạo thêm trạng thái treo. Cụ thể:

- Trạng thái 0: T_0 thông bão hoà, T_1 tắt, $U_{ra} = \text{“L”} \sim 0V$. $Z_{ra0} = Z$ của T_0 bão hoà.

- Trạng thái 1: T_1 thông bão hoà, T_0 tắt, $U_{ra} = \text{“H”} \sim 5V$. $Z_{ra0} = Z$ của T_1 bão hoà.

- Trạng thái treo: T_0 và T_1 đều tắt, $Z_{ra} = \text{vô cùng}$.

Đầu vào, đầu ra không ảnh hưởng lẫn nhau.



Hình 2.8: Sơ đồ tương đương của mạch ra ba trạng thái

Ký hiệu của mạch ra ba trạng thái (hình 2.9)

Tác dụng của mạch ra ba trạng thái:

- Khi ở trạng thái treo, đầu vào không ảnh hưởng đến đầu ra, do vậy có thể sử dụng mạch ra ba trạng thái để tạo nên những đường thông tin hai chiều.

Ví dụ: Sơ đồ sử dụng mạch ra ba trạng thái tạo nên đường thông tin hai chiều cho ở hình 2.10.

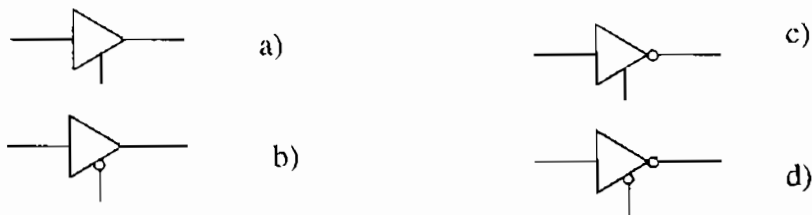
- Nguyên lý làm việc:

- + Khi $X_{dk} = 1$ mạch làm việc bình thường, thông tin đi từ A đến B. Mạch 2 ở trạng thái treo.

- + Khi $X_{dk} = 0$ mạch 2 làm việc bình thường, thông tin đi từ B đến A, mạch 1 ở trạng thái treo.

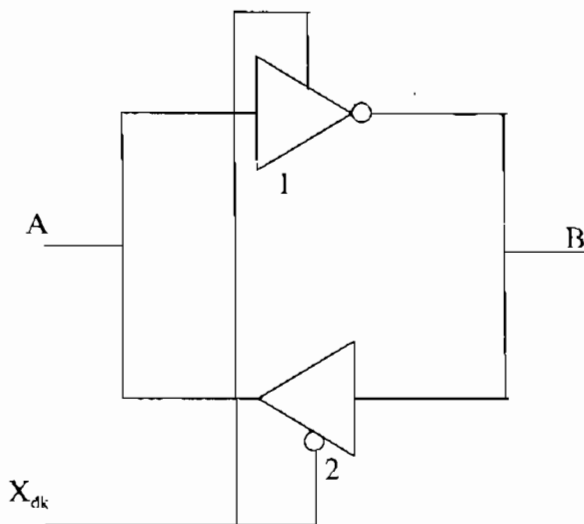
- Nối chung các đầu ra ba trạng thái. Mạch có đầu ra ba trạng thái có thể nối chung các đầu ra với nhau. Chỉ lưu ý phải tạo tín hiệu điều khiển sao cho tại một thời điểm bất kỳ chỉ có duy nhất một mạch hoạt động, còn các mạch khác phải ở trạng thái trở kháng cao. Nếu có hai mạch trở lên cùng làm việc thì có thể xảy ra hỏng mạch như khi nối chung các đầu ra của mạch ra cột chạm.

- Vì mạch có đầu ra ba trạng thái cho phép việc ghép chúng trên BUS chung trở nên dễ dàng.



Hình 2.9. Ký hiệu của mạch ra ba trạng thái

- a. Bộ đệm không đảo với đầu vào điều khiển tích cực H
- b. Bộ đệm không đảo với đầu điều khiển vào tích cực L
- c. Bộ đệm đảo với đầu vào điều khiển tích cực H
- d. Bộ đệm đảo với đầu vào điều khiển tích cực L



Hình 2.10: Tạo thông tin hai chiều từ mạch ra ba trạng thái

III. CÁC PHƯƠNG PHÁP BIỂU DIỄN HÀM LOGIC

Để biểu diễn hàm số logic người ta sử dụng một số phương pháp. Thường dùng 4 phương pháp, đó là bảng chân lý, biểu thức logic, bảng Karnaugh và sơ đồ logic. Chúng ta không những cần nắm vững từng phương pháp, mà còn phải thành thạo chuyển đổi từ phương pháp này sang phương pháp khác.

1. Bảng chân lý

Bảng chân lý là bảng miêu tả quan hệ giữa các giá trị của hàm số tương ứng với mọi giá trị có thể có của biến số.

1.1. Phương pháp liệt kê thành bảng chân lý

Mỗi biến đầu vào có 2 giá trị 1 và 0, nếu có n biến đầu vào thì có 2^n tổ hợp các giá trị khác nhau của chúng. Để nhận được bảng chân lý, ta phải liệt kê tất cả các tổ hợp giá trị của biến đầu vào và giá trị xác định của hàm đầu ra tương ứng với từng tổ hợp đó.

Ví dụ 1: Hãy kê bảng chân lý của hàm số sau:

$$F = AB + BC + CA$$

Giải: Có 3 biến đầu vào, tức là có 8 tổ hợp các giá trị của chúng. Thay giá trị của mỗi tổ hợp vào hàm số và tính ra giá trị tương ứng, rồi liệt kê thành bảng 2.13. (để tránh bỏ sót và khỏi trùng lặp, thường sắp xếp thứ tự các giá trị biến vào theo thứ tự số đếm nhị phân).

Bảng 2.13

C	B	A	F
0	0	0	0
0	0	1	0
0	1	0	0
0	1	1	1
1	0	0	0
1	0	1	1
1	1	0	1
1	1	1	1

Ví dụ 2: Một bóng đèn được điều khiển bằng 3 công tắc. Đèn chỉ sáng khi số lượng công tắc cùng đóng là quá bán. Hãy thiết kế mạch điều khiển trên.

Giải: Gọi A, B, C, là các công tắc đóng, ngắt. Khi công tắc đóng điện thì các biến lấy giá trị 1, ngắt điện thì các biến lấy giá trị 0. Gọi F là trạng thái đèn được điều khiển, đèn sáng $F = 1$, đèn tắt $F = 0$. Sau khi suy xét kỹ, ta kê được bảng chân lí 2.14.

Bảng 2.14

C	B	A	F	Thuyết minh
0	0	0	0	Đèn sáng
0	0	1	0	
0	1	0	0	
0	1	1	1	
1	0	0	0	Đèn sáng
1	0	1	1	
1	1	0	1	
1	1	1	1	

Nếu phải giải quyết một vấn đề logic thực tế, đầu tiên ta hãy làm rõ đâu là đầu vào, đâu là đầu ra, dùng biến đại số biểu thị; tiếp theo cần xác định quan

hệ tương ứng của các trạng thái đầu ra - đầu vào. Cuối cùng liệt kê bảng chân lí một cách chính xác.

1.2. Đặc điểm bảng chân lí

Bảng chân lí biểu thị hàm logic dưới dạng bảng số, nó có các đặc điểm chủ yếu sau đây:

a- Rõ ràng, trực quan. Sau khi xác định giá trị biến đầu vào thì có thể tra bảng chân lí để biết giá trị tương ứng của hàm đầu ra. Vậy nên trong các sổ tay vi mạch số đều có bảng chân lí để giới thiệu chức năng logic của vi mạch.

b- Để giải quyết một nhiệm vụ thực tế ở dạng vấn đề logic, thì bảng chân lí là tiện nhất. Vậy nên trong quá trình thiết kế logic của mạch số, việc đầu tiên là phân tích yêu cầu, kê ra bảng chân lí.

Nhược điểm chủ yếu của bảng chân lí là sẽ rối rắm nếu biến số khá nhiều, không thể dùng các công thức và định lí của đại số logic để tính toán.

Để đơn giản, đôi khi chỉ kê tổ hợp các giá trị đầu vào nào tương ứng hàm số lấy giá trị bằng 1. Những tổ hợp thực tế sử dụng không cần, hoặc làm cho hàm số lấy giá trị 0 đều không cần kê ra.

2. Biểu thức hàm số

Biểu thức hàm số dạng đại số logic dùng các phép toán và, hoặc, đảo biểu thị quan hệ logic giữa các biến trong hàm.

2.1. Dạng chuẩn tắc tuyển (tổng các tích)

Chỉ chú ý đến tổ hợp giá trị các biến nào tương ứng hàm có giá trị 1 trong bảng chân lí. Trong tổ hợp đã chọn, giá trị 1 viết nguyên biến, giá trị 0 viết đảo biến, và kết quả viết được một số hạng dạng tích các biến tương ứng với tổ hợp xét - nếu đem cộng tất cả các số hạng như vậy, thì ta được dạng chuẩn tắc tuyển (tổng các tích - ORAND) của hàm logic.

Ví dụ 1: Hãy viết biểu thức số từ bảng chân lí 2.15.

Giải: Hàm $F = 1$ tương ứng 4 tổ hợp giá trị các biến.

$ABC = 011, 101, 110, 111$. Các số hạng dạng tích các biến là $\overline{A}BC$, $A\overline{B}C$, $AB\overline{C}$, ABC . Dạng chuẩn tắc tuyển của hàm số:

$$F = \overline{A}BC + A\overline{B}C + AB\overline{C} + ABC$$

Kết quả này có chính xác không? Chúng ta có thể nghiệm lại.

Bảng 2.15

A	B	C	F
0	0	0	0
0	0	1	0
0	1	0	0
0	1	1	1
1	0	0	0
1	0	1	1
1	1	0	1
1	1	1	1

Biểu thức hàm số chuẩn tắc tuyển có tên gọi nhấn mạnh hình thức chuẩn của các số hạng dạng tích trong biểu thức. Chúng ta gọi số hạng chuẩn này là số hạng nhỏ nhất.

2.2. Số hạng nhỏ nhất

2.2.1. Định nghĩa

Số hạng nhỏ nhất là một khái niệm quan trọng trong đại số logic. Như ở ví dụ mục 2.1, F là hàm của các biến A, B, C. 3 biến có 8 tổ hợp các giá trị khả dĩ: 000, 001, 010, 011, 100, 101, 110, 111. Tương ứng ta có 8 số hạng dạng tích là:

$$\overline{A} \overline{B} \overline{C}, \overline{A} \overline{B} C, \overline{A} B \overline{C}, \overline{A} B C, A \overline{B} \overline{C}, A \overline{B} C, A B \overline{C}, A B C$$

* Đặc điểm chung của 8 số hạng này là:

- Đều có 3 thừa số.
- Mỗi biến số xuất hiện chỉ 1 lần dưới dạng thừa số hoặc là nguyên biến hoặc là đảo biến.

Vậy chúng ta gọi 8 số hạng dạng tích có đặc điểm trên là số hạng nhỏ nhất của các biến A, B, C.

Nói chung, đối với trường hợp n biến, số hạng tích P có n thừa số; trong P mỗi biến đều xuất hiện một lần, và chỉ 1 lần mà thôi, hoặc dưới dạng nguyên biến, hoặc dưới dạng đảo biến; P được gọi là số hạng nhỏ nhất của n biến, n biến có tất cả 2^n số hạng nhỏ nhất. Vì mỗi biến đều có 2 trạng thái (nguyên biến và đảo biến), mà tất cả có n biến.

2.2.2. Ký hiệu của số hạng nhỏ nhất

Để tiện viết, thường gán cho mỗi số hạng nhỏ nhất một kí hiệu. Phương pháp như sau: tổ hợp các giá trị biến số tương ứng với số hạng nhỏ nhất được

xét, chuyển hình thức số nhị phân sang số thập phân, con số này là kí hiệu của số hạng nhỏ nhất xét.

Ví dụ, trong các số hạng nhỏ nhất của các biến A, B, C thì $\overline{A}\overline{B}\overline{C}$ tương ứng tổ hợp giá trị 000, tức là 0_{10} , kí hiệu của $\overline{A}\overline{B}\overline{C}$ vì vậy là m_0 ; $\overline{A}\overline{B}C$ tương ứng tổ hợp giá trị 010, tức là 2_{10} , kí hiệu của $\overline{A}\overline{B}C$ là m_2 .

Tương tự, $\overline{A}B\overline{C} = m_4$; $\overline{A}BC = m_5$; $A\overline{B}\overline{C} = m_6$; $A\overline{B}C = m_7$. Hơn nữa, thường dùng kí hiệu biểu thị các số hạng nhỏ nhất của dạng chuẩn tắc tuyển.

Ví dụ:

$F = \overline{A}BC + \overline{A}\overline{B}C + A\overline{B}\overline{C} + ABC$ thường viết thành:

$$F = m_3 + m_5 + m_6 + m_7 = \Sigma (3, 5, 6, 7).$$

Tương tự, ta có:

$$F = \overline{A}\overline{B}\overline{C} + ABC = m_0 + m_7 = \Sigma (0, 7)$$

2.3. Dạng chuẩn tắc hội (tích các tổng)

Dạng chuẩn tắc hội có thể nhận được bằng phương pháp sau:

Từ bảng chân lí tìm dạng chuẩn tắc tuyển của đảo hàm, sau đó dùng định lí De Morgan để tìm đảo của đảo hàm.

Ví dụ, từ phần trên ta đã tìm được $\overline{F} = \overline{A}\overline{B}\overline{C} + \overline{A}\overline{B}C + \overline{A}B\overline{C} + A\overline{B}\overline{C}$

$$F = \overline{\overline{F}} = \overline{\overline{A}\overline{B}\overline{C} + \overline{A}\overline{B}C + \overline{A}B\overline{C} + A\overline{B}\overline{C}} = \overline{\overline{A}\overline{B}\overline{C}} \cdot \overline{\overline{A}\overline{B}C} \cdot \overline{\overline{A}B\overline{C}} \cdot \overline{A\overline{B}\overline{C}}$$

$= (A + B + C)(A + B + \overline{C})(A + \overline{B} + C)(\overline{A} + B + C)$ (là biểu thức của hàm Z dạng chuẩn tắc hội).

Các thừa số của hàm số dạng chuẩn tắc hội có tính chất sau:

- Luôn bao gồm tất cả các biến của hàm.
- Mỗi biến đều xuất hiện một lần và chỉ một lần trong dạng tổng của thừa số, hoặc là nguyên biến, hoặc là đảo biến.

Các thừa số có tính chất nêu trên được gọi là thừa số lớn nhất. Tích các thừa số lớn nhất là dạng chuẩn tắc hội của hàm số.

Bảng 2.16

A	B	C	$A+B+C$	$A+B+\overline{C}$	$A+\overline{B}+C$	$A+\overline{B}+\overline{C}$	$\overline{A}+B+C$	$\overline{A}+B+\overline{C}$	$\overline{A}+\overline{B}+C$	$\overline{A}+\overline{B}+\overline{C}$
0	0	0	0	1	1	1	1	1	1	1
0	0	1	1	0	1	1	1	1	1	1
0	1	0	1	1	0	1	1	1	1	1

0	1	1	1	1	1	0	1	1	1	1
1	0	0	1	1	1	1	0	1	1	1
1	0	1	1	1	1	1	1	0	1	1
1	1	0	1	1	1	1	1	1	0	1
1	1	1	1	1	1	1	1	1	1	0

Nói chung, đối với trường hợp hàm n biến, thừa số lớn nhất là một tổng của n số hạng, mỗi số hạng là một biến, xuất hiện một lần dưới dạng nguyên biến hoặc đảo biến, và chỉ xuất hiện một lần mà thôi. n biến có tương ứng 2^n thừa số lớn nhất. Bảng 2.16 là bảng chân lí của toàn bộ các thừa số lớn nhất tương ứng hàm 3 biến A, B, C.

Nhận xét bảng 2.16, ta thấy thừa số lớn nhất có các tính chất sau:

- Mỗi thừa số lớn nhất tương ứng với một tổ hợp giá trị của biến để nó bằng 0, và chỉ có một tổ hợp mà thôi.
- Tổng của hai thừa số lớn nhất bất kì luôn luôn bằng 1.
- Tích của tất cả các thừa số lớn nhất luôn bằng 0.

Cách kí hiệu các thừa số lớn nhất như sau: Tổ hợp các giá trị biến số tương ứng với thừa số lớn nhất được xét chuyển hình thức số nhị phân sang số thập phân, con số này là kí hiệu của thừa số lớn nhất xét. Ví dụ, trong các thừa số lớn nhất của các biến A, B, C thì:

A+B+C tương ứng tổ hợp	000	chuyển thành	0_{10}	kí hiệu	M_0
$A+B+\bar{C}$	001		1_{10}		M_1
$A+\bar{B}+C$	010		2		M_2
$A+\bar{B}+\bar{C}$	011		3		M_3
$\bar{A}+B+C$	100		4		M_4
$\bar{A}+B+\bar{C}$	101		5		M_5
$\bar{A}+\bar{B}+C$	110		6		M_6
$\bar{A}+\bar{B}+\bar{C}$	111		7		M_7

Cách viết kí hiệu rất thuận tiện. Chú ý rằng m_i và M_i là đảo của nhau:

$$m_i = \overline{M_i}$$

$$\text{Ví dụ: } m_0 = \overline{ABC} \quad \overline{M_0} = A + B + C$$

$$m_0 = M_0 = \overline{A + B + C} = \overline{ABC}$$

$$m_5 = \overline{ABC} \quad M_5 = \overline{\bar{A} + B + \bar{C}}$$

$$m_5 = M_5 = \overline{A + B + C} = \overline{A} \overline{B} \overline{C}$$

Thừa số lớn nhất cũng là phần tử cơ bản cấu trúc hàm logic. Biểu thức hàm số $F = (A + B + C) (A + B + \overline{C}) (A + \overline{B} + C) (\overline{A} + B + C)$ có thể viết dưới dạng:

$$Z = M_0 M_1 M_2 M_4 = \Pi (0, 1, 2, 4)$$

Đặc điểm các biểu thức hàm số:

Một hàm số logic được biểu thị bằng biểu thức các phép toán và, hoặc, đảo v.v... liên kết các biến số của nó với nhau. Ưu điểm của phương pháp biểu thức hàm số là:

- Dùng các kí hiệu logic biểu thị quan hệ logic giữa các biến làm cho cách viết gọn và tiện, tính khái quát và trừu tượng rất cao.
- Rất tiện sử dụng các công thức và định lí của đại số logic để biến đổi, làm toán.
- Tiện cho việc dùng sơ đồ logic để thực hiện hàm số. Chỉ cần dùng các kí hiệu logic của mạch điện cổng tương ứng thay thế phép toán xét trong biểu thức hàm số, ta được một sơ đồ logic. Vấn đề này còn được giới thiệu cụ thể sau.

Nhược điểm chủ yếu của phương pháp biểu thức hàm số là khó xác định giá trị hàm ứng với giá trị biến một cách trực tiếp đối với các hàm số phức tạp (không trực quan như bảng chân lí).

3. Bảng Karnaugh

Bảng Karnaugh là phương pháp hình vẽ biểu thị hàm logic, trong đó các giá trị hàm đầu ra tương ứng tổ hợp các biến đầu vào đều được biểu thị đầy đủ. Trên cơ sở bảng Karnaugh của các biến, điền các số hạng nhỏ nhất của hàm số vào các ô tương ứng thì ta có bảng Karnaugh của hàm.

3.1. Bảng Karnaugh của biến logic

3.1.1. Bảng Karnaugh 3 biến và 4 biến (Hình 2.11 và 2.12)

A \ BC				
	00	01	11	10
0	m_0	m_1	m_3	m_2
1	m_4	m_5	m_7	m_6

a) Ghi ký hiệu số hạng nhỏ nhất trong ô

A \ BC				
	00	01	11	10
0				
1				

b) Không ghi số hạng nhỏ nhất

Hình 2.11: Bảng Karnaugh cho hàm 3 biến (A,B,C)

3.1.2. Quy tắc vẽ bảng Karnaugh của biến

- Bảng Karnaugh có dạng hình chữ nhật, khi hàm có n biến sẽ có 2^n ô, mỗi ô tương ứng với một số hạng nhỏ nhất. Ví dụ, hình 3-2-1, $n = 3$ tương ứng bảng

$2^3 = 8$ ô, $n = 4$ tương ứng bảng $2^4 = 16$ ô. Một hàm số có n biến, nếu lấy k biến để biểu diễn các cột sẽ được 2^k cột. (k là số nguyên, dương và nhỏ hơn n). Số biến còn lại để biểu diễn hàng là $n - k$ và như vậy sẽ có 2^{n-k} hàng.

CD \ AB	CD			
	00	01	11	10
00				
01				
11				
10				

a) Không ghi ký hiệu số hạng nhỏ nhất trong ô

CD \ AB	CD			
	00	01	11	10
00	0	1	3	2
01	4	5	7	6
11	12	13	15	14
10	8	9	11	10

b) Ghi số hạng nhỏ nhất

Hình 2.12: Bảng Karnaugh cho hàm 4 biến(A,B,C,D)

- Giá trị các biến được sắp xếp thứ tự theo mã Gray. (Nếu không sắp xếp thứ tự theo mã Gray thì không còn là bảng Karnaugh nữa!).

Ví dụ: Sự sắp xếp của AB và CD đều là 00, 01, 11, 10 (hình 2.12).

3.1.3. Đặc điểm bảng Karnaugh của biến

- Ưu điểm lớn nhất của bảng là làm nổi bật tính kề nhau của các số hạng nhỏ nhất. Các ô kề nhau bất kì trên bảng đều có các số hạng nhỏ nhất đều có tính kề nhau về logic. Sự sắp xếp giá trị các biến theo mã vòng bảo đảm đặc điểm quan trọng này.

Tính kề nhau bao gồm 3 tình huống sau: Các ô kề nhau, các ô đầu cuối của hàng và cột, các ô đối xứng đều phải có một giá trị đối nhau của biến và chỉ một mã thôi. Đặc điểm này của bảng cho phép dễ dàng nhớ và phân biệt, kiểm tra, tính toán bằng bảng, nhất là khi có đến 5, 6 biến.

Như trên đã nói, nếu trong hai số hạng nhỏ nhất có và chỉ có 1 biến lấy giá trị khác nhau, còn tất cả các biến khác đều lấy giá trị như nhau, thì hai số hạng nhỏ nhất đó có tính kề nhau về logic. Ví dụ, trong hình 2.11, m_0 có tính kề nhau về logic với m_1 , m_2 và m_4 .

Khi cộng các số hạng nhỏ nhất có tính kề nhau, thì biến đối nhau trong đó sẽ bị khử.

Ví dụ: $m_0 + m_1 = \overline{A}\overline{B}\overline{C} + \overline{A}\overline{B}C = \overline{A}\overline{B}(\overline{C} + C) = \overline{A}\overline{B}$; $\overline{A}\overline{B}$ là thừa số chung của m_0 và m_1 ; $m_0 + m_2 = \overline{A}\overline{B}\overline{C} + \overline{A}B\overline{C} = \overline{A}\overline{C}$ khử mất B và $\overline{B}$; $m_0 + m_4 = \overline{A}\overline{B}\overline{C} + \overline{A}B\overline{C} = \overline{A}\overline{C}$ khử mất B và $\overline{B}$.

- Nhược điểm chủ yếu của bảng Karnaugh: Nếu số biến tăng thì độ phức tạp của bảng tăng nhanh. Ví dụ, nếu số biến từ 7 trở đi thì hình vẽ quá phức tạp, hơn nữa rất khó xét đoán tính kế nhau về logic của các số hạng nhỏ nhất. Vì vậy, bảng Karnaugh chỉ thích hợp để biểu thị hàm logic có số biến từ 6 trở lại.

3.2. Bảng Karnaugh của hàm logic

3.2.1. Các cách biểu diễn

Có 3 trường hợp:

Trường hợp thứ nhất: Đã cho bảng chân lí của hàm.

Trên bảng Karnaugh của biến, điền giá trị 1 vào ô mà hàm lấy giá trị 1 tương ứng tổ hợp giá trị các biến của ô xét, điền giá trị 0 vào ô mà hàm lấy giá trị 0 tương ứng tổ hợp giá trị các biến của ô xét.

Bảng 2.17

Ví dụ: Cho bảng chân lí 2.17 (hình bên).

Hãy vẽ bảng Karnaugh của hàm F.

- Đầu tiên vẽ bảng Karnaugh cho 4 biến A, B, C, D.

- Tiếp theo điền các giá trị của hàm F vào các ô tương ứng hợp với bảng chân lí.

- Kết quả: Hình 2.13.

A	B	C	D	F
0	0	0	0	0
0	0	0	1	0
0	0	1	0	0
0	0	1	1	0
0	1	0	0	0
0	1	0	1	0
0	1	1	0	0
0	1	1	1	1
1	0	0	0	0
1	0	0	1	0
1	0	1	0	0
1	0	1	1	1
1	1	0	0	0
1	1	0	1	1
1	1	1	0	1
1	1	1	1	1

Trường hợp thứ hai: Đã cho biểu thức của hàm dưới dạng chuẩn tắc tuyển. Trên bảng Karnaugh của biến, điền giá trị 1 vào ô tương ứng với từng số hạng nhỏ nhất có trong biểu thức, các ô khác đều điền vào giá trị 0.

Ví dụ:

Hãy vẽ bảng Karnaugh của hàm logic:

$$F(A,B,C,D) = \Sigma (0, 3, 5, 6, 9, 10, 12, 15)$$

Giải:

- Vẽ bảng Karnaugh của 4 biến A, B, C, D.
- Điền giá trị cặp biến và giá trị của hàm tại các ô có tổ hợp biến tính theo giá trị ở hệ mười là 0, 3, 5, 6, 9, 10, 12 và 15.

Kết quả: hình 2.14.

AB \ CD	00	01	11	10
00	0	0	0	0
01	0	0	1	0
11	0	1	1	1
10	0	0	1	0

Hình 2.13: Biểu diễn theo bảng 2.17

AB \ CD	00	01	11	10
00	1	0	1	0
01	0	1	0	1
11	1	0	1	0
10	0	1	0	1

Hình 2.14: Biểu diễn theo ví dụ ở trường hợp thứ hai

Trường hợp thứ ba: Cho biểu thức không chuẩn tắc của hàm.

Biến đổi hàm đã cho thành dạng tổng các tích.

Trên bảng Karnaugh của biến, điền giá trị 1 vào tất cả các ô tương ứng số hạng nhỏ nhất bao hàm trong số hạng dạng tích nói trên, sau đó điền giá trị 0 vào các ô còn lại.

Ví dụ: Hãy vẽ bảng Karnaugh của hàm $F = \overline{(A \oplus B)}(C + D)$

Giải:

- Biến đổi hàm thành dạng tổng các tích:

$$\begin{aligned} F &= \overline{(A \oplus B)}(C + D) = \overline{A \oplus B} + \overline{C + D} \\ &= \overline{A \oplus B} + \overline{C} \overline{D} \end{aligned}$$

- Xác định mỗi số hạng bao gồm những số hạng nhỏ nhất nào:

$$\overline{A \oplus B} = m_0 + m_1 + m_2 + m_3$$

$$\overline{C} \overline{D} = m_{12} + m_{13} + m_{14} + m_{15}$$

$$\overline{C} \overline{D} = m_0 + m_4 + m_8 + m_{12}$$

- Kết quả vẽ được như hình 2.15.

AB \ CD	00	01	11	10
00	1	1	1	1
01	1	0	0	0
11	1	1	1	1
10	1	0	0	0

Hình 2.15

3.2.2. Từ bảng Karnaugh kê ra bảng chân lí và viết biểu thức

Bảng chân lí, hàm dạng chuẩn tắc tuyển và bảng Karnaugh đều là duy nhất biểu thị cho một hàm, chúng tất có quan hệ chuyển đổi nhau. Thực tế ở các phần trên đã chuyển đổi rồi.

Ưu điểm nổi bật nhất của bảng Karnaugh là tính kế nhau về logic của các số hạng nhỏ nhất của hàm biểu thị rõ rệt thành sự liên kế hình học của các ô trong bảng, do đó dễ dàng tối thiểu hoá hàm đã cho. Vấn đề này sẽ trình bày chi tiết ở phần sau.

4. Sơ đồ logic

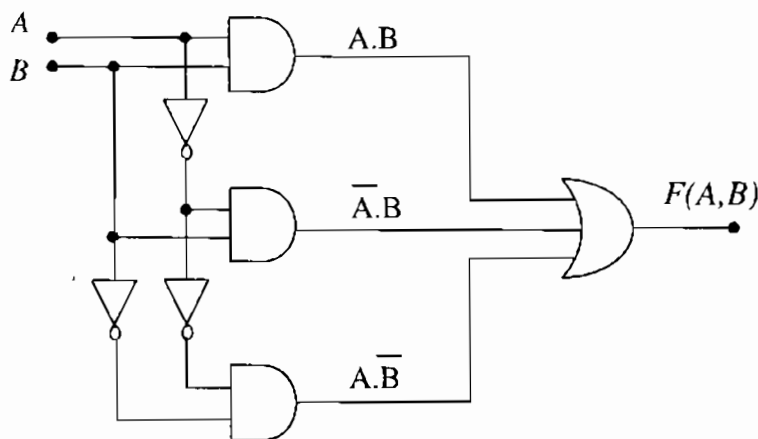
Trong mạch số, sau khi dùng các ký hiệu logic biểu thị một cấu trúc logic trên một bản vẽ, ta được sơ đồ logic. Sơ đồ logic cũng là một phương pháp biểu thị hàm logic, hơn nữa lại có ưu điểm nổi bật là rất tiếp cận thực tế. Các ký hiệu logic thông thường đều có cấu kiện điện tử cụ thể tương ứng, vậy nên thường gọi sơ đồ logic là sơ đồ mạch logic.

4.1. Cách vẽ sơ đồ logic của hàm logic

Như trên đã nói, ta dùng ký hiệu logic của mạch điện tử thay thế phép tính logic có trong biểu thức hàm logic thì được sơ đồ logic của hàm.

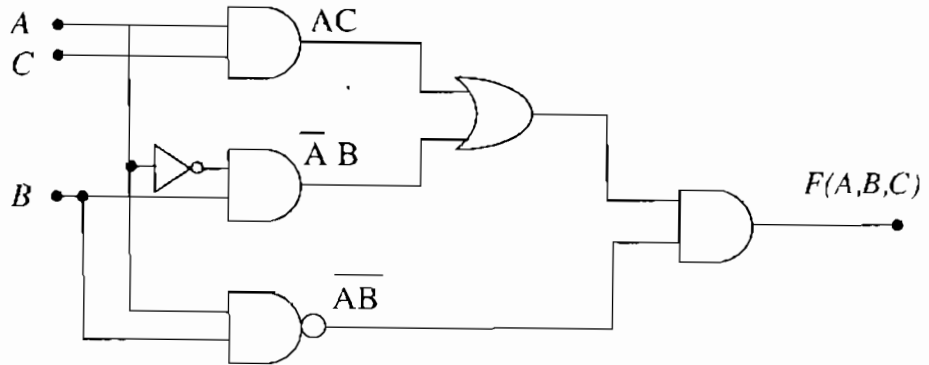
Ví dụ 1: Cho hàm $F = AB + \bar{A}B + A\bar{B}$. Hãy vẽ sơ đồ logic của hàm.

Giải: Quan hệ nhân logic của các biến A và B , A đảo và B , A và B đảo được thực hiện bằng các cổng AND. Quan hệ cộng logic của các số hạng AB , $\bar{A}B$ và $A\bar{B}$ được thực hiện bằng cổng OR. Thực hiện phủ định A có $\bar{A}$ đảo, phủ định B có $\bar{B}$ đảo. Kết quả vẽ được hình 2.16:



Hình 2.16: Sơ đồ logic của hàm vẽ theo ví dụ 1

Ví dụ 2: Cho hàm số $F = (AC + \bar{A}B)(\bar{A}B)$, vẽ sơ đồ logic của hàm.



Hình 2.17: Sơ đồ logic của hàm $F = (AC + \bar{A}B)(\bar{A}B)$

- Quan hệ AC là quan hệ nhân logic, $\bar{A}B$ là quan hệ nhân logic của A đảo với B sau khi đã đảo A .
- Sau khi thực hiện các quan hệ trên dùng quan hệ cộng logic (*).
- Quan hệ $\bar{A}B$ là quan hệ nhân logic sau đó đảo kết quả của phép nhân (**).
- Từ đây lại thực hiện một phép nhân logic thực hiện quan hệ giữa (*) và (**).

Thực hiện các quan hệ trên bằng các phần tử logic ta có sơ đồ như hình 2.17.

4.2. Đặc điểm của sơ đồ logic

Các kí hiệu logic trong sơ đồ logic có quan hệ phù hợp rõ ràng với cấu kiện điện tử trong thực tế, vậy sơ đồ logic tương đối tiếp cận thực tế công trình. Trong công tác, khi tìm hiểu chức năng logic của một hệ thống số nào đó hay là thiết bị được điều khiển số nào đó, thường ta cần dùng sơ đồ logic, vì rằng sơ đồ logic có thể biểu thị rõ ràng chức năng logic từng tầng của các mạch điện thực tế phức tạp. Trong việc chế tạo các thiết bị số, việc đầu tiên là thiết kế logic để vẽ ra sơ đồ logic, rồi chuyển từ sơ đồ logic thành mạch điện thực tế.

IV. KHÁI NIỆM VỀ TỐI THIỂU HOÁ VÀ CÁC PHƯƠNG PHÁP TỐI THIỂU

1. Khái niệm về tối thiểu hoá

Dưới đây sẽ nói đến khái niệm tối thiểu hoá, tiếp theo sẽ giới thiệu 2 phương pháp thường dùng để tối thiểu hoá.

Tối thiểu hoá là dùng các phương pháp để biến đổi một hàm logic từ dạng phức tạp về dạng đơn giản mà không làm thay đổi bản chất của hàm.

1.1. Các loại biểu thức logic và sự thực hiện bằng mạch điện

Ta đã biết, một hàm logic có thể có rất nhiều biểu thức khác nhau. Chúng ta có thể phân loại theo các hàm logic thành 5 loại căn cứ vào đặc điểm và quan hệ giữa các số hạng dạng tích trong hàm: OR-AND, AND-OR, NAND-NAND, NOR-NOR, NOR-AND. Ví dụ:

$F = AB + \bar{A}C$	dạng biểu thức	OR-AND
$F = \overline{(A + C)} (\bar{A} + B)$		AND-OR
$F = \overline{\overline{A} \overline{B} \overline{A} \overline{B}}$		NAND - NAND
$F = \overline{\overline{A + C} + \overline{\bar{A} + B}}$		NOR-NOR
$F = \overline{A \cdot B + A \cdot C}$		NOR-AND

Khi dùng các cổng logic thực hiện các hàm logic này, tiện nhất là: hai loại đầu dùng cổng AND và cổng OR, loại thứ ba dùng NAND, loại thứ tư dùng NOR, loại thứ năm dùng NOR-AND.

Trên thực tế, khi chúng ta viết một hàm logic dưới một dạng nào đó, thì dạng có được không phải là duy nhất. Ví dụ, biểu thức OR-AND trong ví dụ trên có thể viết thành:

$$\begin{aligned}
 F &= AB + \bar{A}C \\
 &= AB + \bar{A}C + BC \quad (\text{theo công thức 15}) \\
 &= ABC + ABC + \bar{A}BC + \bar{A}BC \\
 &= \dots
 \end{aligned}$$

Dùng các cổng AND và OR thực hiện $F = AB + \bar{A}C$ ta có mạch đơn giản nhất. Nói chung, nếu biểu thức càng đơn giản thì mạch điện cũng càng đơn giản. Nhưng đối với các biểu thức dạng khác nhau, thì tiêu chuẩn về sự đơn giản có khác nhau. Ta sẽ làm rõ điều này qua ví dụ về biểu thức OR-AND.

1.2. Biểu thức OR-AND tối thiểu

1.2.1. Thế nào là tối thiểu

- Đầu tiên số các số hạng dạng tích phải là ít nhất.
- Nếu điều kiện trên đã bảo đảm, thì số biến của mỗi số hạng cũng phải là ít nhất.

$$\begin{aligned}
 \text{Ví dụ: } F &= \overline{AC} + \overline{BC} + \bar{A}B + \bar{A}C \\
 &= \overline{AC} + \overline{BC} + \bar{A}C
 \end{aligned}$$

$\bar{A}B$ được khử bỏ theo quy tắc của công thức 14. Rõ ràng sau khi tối thiểu hàm đơn giản hơn, vì nó ít hơn một số hạng.

1.2.2. Ý nghĩa việc tối thiểu hoá biểu thức OR-AND

Chúng ta tập trung nghiên cứu phương pháp tối thiểu hoá biểu thức OR-AND, vì chỉ cần có biểu thức OR-AND tối thiểu, ta sẽ dễ dàng có được các biểu thức dạng khác cũng là tối thiểu. Có hai lí do: một là, một biểu thức logic bất kì đều dễ dàng triển khai thành biểu thức dạng OR-AND; hai là, từ biểu thức dạng OR-AND tối thiểu, cũng dễ dàng có được các biểu thức tối thiểu dạng NAND-NAND, NOR-AND v.v.

2. Sự cần thiết phải tối thiểu hoá

Trực tiếp thiết kế sơ đồ mạch logic theo hàm số có được từ bảng chân lí thường là rất phức tạp. Nếu sau khi đã thực hiện tối thiểu hoá hàm logic, nói chung việc thực hiện thuận tiện hơn, không những chỉ dùng số cấu kiện ít hơn, mà nâng cao được độ tin cậy

3. Các phương pháp tối thiểu cơ bản

Có hai phương pháp cơ bản để tối thiểu hàm số logic. Đó là phương pháp đại số và phương pháp hình học. Dưới đây giới thiệu về hai phương pháp tối thiểu trên.

V. PHƯƠNG PHÁP TỐI THIỂU HOÁ BẰNG CÔNG THỨC

1. Định nghĩa

Dựa vào các công thức và định lí có trong đại số logic để biến đổi hàm logic từ dạng phức tạp về dạng đơn giản mà không làm thay đổi bản chất của hàm. Phương pháp này còn có tên gọi là tối thiểu hoá bằng phương pháp đại số.

Vì trong thực tế, biểu thức logic rất đa dạng, lại không có một cách nào hoàn chỉnh như một quy trình, nên việc đạt đến một biểu thức logic tối thiểu một cách nhanh nhất sẽ hoàn toàn phụ thuộc kinh nghiệm, hiểu biết và thành thạo của chúng ta.

2. Một số ví dụ

* Ví dụ 1: Hãy tối thiểu hoá hàm $F = A\bar{B} + B\bar{C} + \bar{B}C + \bar{A}B$

Giải: $F = A\bar{B} + B\bar{C} + \bar{B}C + \bar{A}B$

$$F = (A\bar{B} + B\bar{C} + A\bar{C} + AC) + (\bar{B}C + \bar{A}B)$$

$$F = (A\bar{B} + B\bar{C} + AC) + (\bar{B}C + \bar{A}B + AC)$$

$$F = (\overline{B}A + \overline{B}C + A\overline{C}) + (B\overline{C} + \overline{A}B + A\overline{C})$$

$$F = (\overline{B}C + A\overline{C}) + (\overline{A}B + A\overline{C})$$

$$F = (\overline{B}C + A\overline{C}) + (A\overline{C} + A\overline{C})$$

$$F = \overline{B}C + \overline{A}B + A\overline{C}$$

Cách giải trên đây ứng dụng công thức 14, thêm vào và bớt đi rất linh hoạt. Chẳng hạn nếu thêm vào hay bớt đi $\overline{A}C$, thì hàm đã có ở ví dụ 1 có dạng tối thiểu hoá mới:

$$F = A\overline{B} + \overline{B}C + \overline{B}C + \overline{A}B = A\overline{B} + \overline{B}C + \overline{A}C$$

Theo công thức 11', ta có thể chỉ viết 1 số hạng $A\overline{C}$ nhưng ngầm hiểu có thể gộp nó vào nhiều nhóm. Trong ví dụ này, thêm vào chỉ một, nhờ gộp vào hai nhóm, nên bớt đi hai, kết quả là khử bớt 1 số hạng. Trong trường hợp tổng quát, ta thường ứng dụng nhiều công thức và định lý để có thể tối thiểu hoá một hàm phức tạp.

* Ví dụ 2: Hãy tối thiểu hoá hàm:

$$F = AD + \overline{A}\overline{D} + AB + \overline{A}C + BD + \overline{A}CEF + \overline{B}EF + DEFG$$

Giải:

- Dùng công thức: $AD + \overline{A}\overline{D} = A(D + \overline{D}) = A$ ta có:

$$F = A + AB + \overline{A}C + BC + \overline{A}CEF + \overline{B}EF + DEFG$$

- Dùng công thức: $A + AB = A$ và $\overline{A}C + \overline{A}CEF = \overline{A}C$ ta có:

$$F = A + \overline{A}C + BD + \overline{B}EF + DEFG$$

- Dùng công thức 13: Khử bỏ $\overline{A}$ trong số hạng $\overline{A}C$:

$$F = A + C + BD + \overline{B}EF + DEFG$$

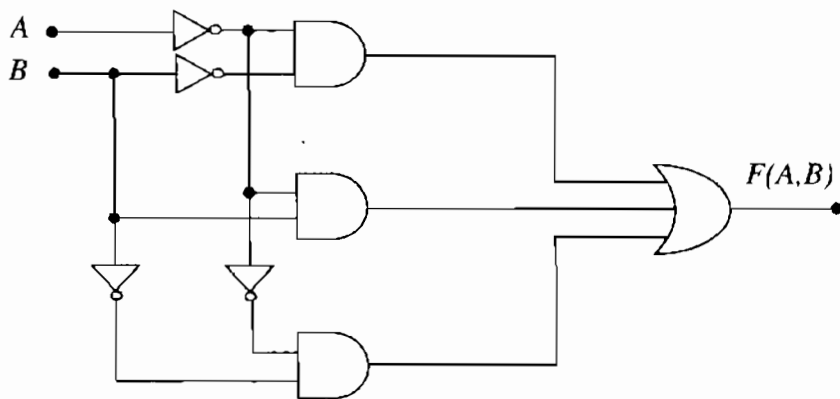
- Dùng công thức 14: Khử bỏ DEFG, vậy:

$$F = A + C + BD + \overline{B}EF$$

* Ví dụ 3: Cho hàm số $F = \overline{A}\overline{B} + \overline{A}B + A\overline{B}$. Vẽ sơ đồ logic của hàm trước khi tối thiểu, tối thiểu hàm và vẽ lại sơ đồ theo hàm đã tối thiểu.

Giải:

- Từ hàm số trên ta thấy có 3 quan hệ AND, 1 quan hệ OR của 3 biến, quan hệ đảo A và đảo B. Sơ đồ vẽ được như hình 2.18.



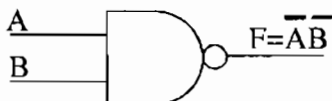
Hình 2.18: Sơ đồ hàm logic vẽ theo ví dụ 3 trước khi tối thiểu.

- Tối thiểu hàm: $F = \overline{A}\overline{B} + \overline{A}B + A\overline{B}$

$$F = \overline{A}(\overline{B} + B) + A\overline{B}$$

$F = \overline{A} + A\overline{B} = \overline{A} + \overline{B}$, theo định lý De Morgan ta có: $F = \overline{A\overline{B}} \Rightarrow$ Đây là hàm NAND hai biến vào là A và B.

Vậy sơ đồ của hàm sau tối thiểu là mạch cổng NAND hai đầu vào như hình 2.19.



Hình 2.19: Sơ đồ logic $F = \overline{A\overline{B}}$ sau khi rút gọn hàm số

VI. PHƯƠNG PHÁP TỐI THIỂU HOÁ BẰNG BẢNG KARNAUGH

Bảng Karnaugh ngoài việc dùng để biểu diễn hàm logic nó còn có thể dùng để tối thiểu hàm logic. Ta sẽ tìm hiểu việc tối thiểu hàm logic bằng bảng Karnaugh theo các nội dung sau.

1. Phương pháp tối thiểu hoá hàm logic được biểu diễn bằng bảng Karnaugh

Một cách tổng quát ta có thể cực tiểu hoá một hàm logic được biểu diễn bằng Karnaugh theo thủ tục sau:

1. Khoanh vòng các ô không thể kết hợp với bất kỳ ô nào khác. Minterm tương ứng với các ô này sẽ được chấp nhận ở kết quả.

2. Xác định các ô có thể kết hợp với chỉ một ô khác theo một cách duy nhất. Khoanh vòng các tổ hợp hai ô này lại với nhau. Các ô có thể kết hợp với một ô khác theo nhiều cách khác nhau thì tạm thời có thể bỏ qua.

3. Xác định các ô có thể kết hợp với ba ô khác theo một cách duy nhất. Nếu tất cả bốn ô kết hợp như vậy không bao trùm hết các nhóm hai ô thì ta khoanh vòng các nhóm bốn ô này lại. Các ô có thể gộp lại để thành nhóm bốn ô theo nhiều cách thì tạm thời cũng bị bỏ qua.

4. Lập lại các bước trên đối với nhóm tám ô, v.v.

5. Sau các bước trên, nếu vẫn còn một số ô chưa được khoanh thì chúng có thể kết hợp với nhau hoặc kết hợp với các ô đã được khoanh một cách tùy ý. Nhưng dĩ nhiên là ta cần kết hợp chúng sao cho số nhóm là ít nhất.

Để làm rõ hơn thủ tục trên ta sẽ xét thêm hai bài toán sau. Bài toán thứ nhất minh họa một lời giải không xét đến bước 5 còn bài toán thứ hai thì có xét đến.

2. Các ví dụ

* Ví dụ 1

Cho hàm logic bốn biến $F(A,B,C,D)$, A có trọng số lớn nhất, D có trọng số nhỏ nhất như sau:

$$F(A,B,C,D) = \sum (0,1,3,5,6,9,11,12,13,15)$$

Hãy cực tiểu hoá hàm này bằng cách dùng bảng Karnaugh.

Lời giải

Bảng Karnaugh biểu diễn hàm ở ví dụ 1 có thể vẽ trên hình 2.20.a.

Ta thực hiện theo các bước của thủ tục trên như sau:

+ Ô m_6 không thể kết hợp với bất cứ ô nào khác. Do đó, ta khoanh vòng nó lại.

+ Các ô m_0 và m_{12} chỉ có thể kết hợp với chỉ một ô khác theo một cách duy nhất, ta khoanh các tổ hợp này lại tạo thành các nhóm hai ô như trên hình 2.20.b. Các ô khác có thể kết hợp để tạo thành nhóm hai ô theo nhiều cách, tạm thời bỏ qua chưa đề cập đến.

+ Các ô m_3 , m_5 và m_{15} có thể tham gia vào các nhóm bốn ô theo một cách duy nhất, ngoài ra ta cũng thấy rằng các nhóm bốn ô kết hợp như vậy không bao trùm hết các nhóm hai ô nên ta khoanh vòng các nhóm bốn ô này lại như trên hình 2.20.c.

+ Cuối cùng ở trên hình 2.20.d ta thấy tất cả các ô đã được khoanh vòng. Từ đó ta được kết quả:

$$F(A,B,C,D) = \overline{A}BC\overline{D} + \overline{A}\overline{B}\overline{C} + A\overline{B}\overline{C} + \overline{C}D + \overline{B}D + \overline{B}D + AD$$

AB \ CD		00	01	11	10
CD	00	1		1	
	01	1	1	1	1
	11	1		1	1
	10		(1)		

a)

AB \ CD		00	01	11	10
CD	00	(1)		(1)	
	01	(1)	1	(1)	1
	11	1		1	1
	10		(1)		

b)

AB \ CD		00	01	11	10
CD	00	1		1	
	01	(1)	1	(1)	(1)
	11	(1)		(1)	(1)
	10		1		

c)

AB \ CD		00	01	11	10
CD	00	(1)		(1)	
	01	(1)	1	(1)	(1)
	11	(1)		(1)	(1)
	10		(1)		

d)

Hình 2.20: Bảng Karnaugh đối với ví dụ 1

* Ví dụ 2

Cho hàm logic bốn biến $F(A,B,C,D)$ như sau:

$$F(A,B,C,D) = \sum(0,2,3,4,5,7,8,9,13,15)$$

Hãy cực tiểu hoá hàm này bằng cách dùng bảng Karnaugh.

Lời giải

Bảng Karnaugh biểu diễn hàm ví dụ 2 được vẽ trên hình 2.21.a.

Ta cũng thực hiện theo các bước của thủ tục trên:

+ Ở bước 1 và bước 2 ta nhận thấy tất cả các ô đều không thoả mãn các điều kiện nên ta không có ô độc lập (không kết hợp được với các ô khác) và các nhóm hai ô.

+ Bốn ô m_5 , m_7 , m_{13} và m_{15} thoả mãn các điều nêu ra ở bước 3, ta khoanh các tổ hợp của những ô này lại như trên hình 2.21.b.

+ Bước 4 không được áp dụng trong trường hợp này.

+ Đối với các ô còn lại, như đòi hỏi ở bước 5, ta gộp chúng lại một cách tùy ý. Để thấy rằng phương án gộp các ô này lại như biểu diễn trên hình 2.21.c là có số nhóm ít nhất.

+ Cuối cùng, từ bảng Karnaugh trên hình 2.21.c ta có thể viết biểu thức của hàm đã cực tiểu hoá như sau:

$$F(A,B,C,D) = \overline{A}\overline{C}\overline{D} + \overline{A}\overline{B}C + A\overline{B}\overline{C} + BD$$

AB \ CD		00	01	11	10
CD	00	1	1		1
	01		1	1	1
	11	1	1	1	
	10	1			

a)

AB \ CD		00	01	11	10
CD	00	1	1		1
	01		1	1	1
	11	1	1	1	
	10	1			

b)

AB \ CD		00	01	11	10
CD	00	1	1		1
	01		1	1	1
	11	1	1	1	
	10	1			

c)

Hình 2.21: Bảng Karnaugh đối với ví dụ 2

3. Một số vấn đề cần lưu ý

- Vòng gộp phải càng to càng tốt. Tương ứng số các số hạng nhỏ nhất được gộp lại càng nhiều; do đó, sau khi gộp, số hạng càng ít thừa số.

- Mỗi vòng gộp bao gồm ít nhất một số hạng nhỏ nhất không có trong vòng khác. Vòng nào bao gồm các số hạng đều đã có trong vòng khác, thì vòng đó là thừa. Mặt khác, mỗi số hạng nhỏ nhất có thể được sử dụng nhiều lần (có mặt trong nhiều vòng khác nhau).

- Phải khoanh vòng sao cho toàn bộ số hạng nhỏ nhất của hàm số đều có các vòng, không sót. Các thừa số tương ứng của số hạng vòng gộp xét làm thành số hạng của hàm đã tối thiểu hoá.

- Trong một số trường hợp, có thể có nhiều cách khoanh vòng, nghĩa là có thể có nhiều hàm tối thiểu. Nhưng hàm tối thiểu này cần phải được so sánh, kiểm tra để chọn ra đâu là hàm tối thiểu thực sự (tối thiểu của tối thiểu)

- Khi gộp các số hạng nhỏ nhất, nghĩa là khi khoanh vòng, có 2 điều sau đây dễ quên: Một là, phải nhớ rằng 4 ô ở 4 góc bảng Karnaugh cũng có thể gộp với nhau; hai là, vẽ vòng lớn trước, vòng bé sau, kiểm tra xem: mỗi vòng có ít nhất một số hạng nhỏ nhất không có trong vòng khác. Không lưu ý đến những vấn đề này, biểu thức hàm số đạt được không chắc là tối thiểu.

VII. MẠCH TỔ HỢP

1. Khái niệm

Căn cứ vào đặc điểm và chức năng logic, chúng ta phân loại mạch số thành hai loại chính: mạch tổ hợp và mạch dãy (mạch dãy được trình bày trong chương 6, 7 và 8).

1.1. Đặc điểm cơ bản của mạch tổ hợp

Trong mạch số, mạch tổ hợp là mạch mà trị số ổn định của tín hiệu đầu ra ở thời điểm bất kì chỉ phụ thuộc vào tổ hợp các giá trị tín hiệu đầu vào ở thời điểm đó. Trong mạch tổ hợp, trạng thái mạch điện trước thời điểm xét, tức trước khi có tác động của tín hiệu đầu vào, không ảnh hưởng đến tín hiệu đầu ra. Đặc điểm cấu trúc mạch tổ hợp là được cấu trúc nên từ các cổng logic. Vậy các mạch điện cổng ở chương 2 và các mạch logic sẽ gặp ở chương 3, 4 và 5 đều là mạch tổ hợp.

1.2. Phương pháp biểu thị chức năng logic

Các phương pháp thường dùng để biểu thị chức năng logic của mạch tổ hợp là hàm số logic, bảng chân lí, sơ đồ logic, bảng Karnaugh, cũng có khi biểu thị bằng đồ thị thời gian dạng sóng.

Đối với vi mạch cỡ nhỏ (SSI) thường biểu thị bằng hàm logic. Đối với vi mạch cỡ vừa thường biểu thị bằng bảng chân lí, hay là bảng chức năng. Bảng chức năng dùng hình thức bảng kê, với mức logic cao (H) và mức logic thấp (L), để mô tả quan hệ logic giữa tín hiệu đầu ra với tín hiệu đầu vào của mạch điện xét. Chỉ cần thay giá trị logic cho trạng thái trong bảng chức năng, thì ta có bảng chân lí tương ứng.

Như hình 2.22 cho biết, thường có nhiều tín hiệu đầu vào và nhiều tín hiệu đầu ra. Một cách tổng quát, hàm logic của tín hiệu đầu ra có thể viết dưới dạng:

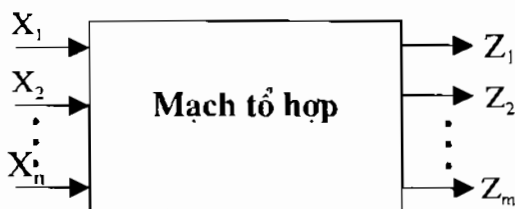
$$z_1 = f_1(x_1, x_2, \dots, x_n)$$

$$z_2 = f_2(x_1, x_2, \dots, x_n)$$

$$z_m = f_m(x_1, x_2, \dots, x_n)$$

Cũng có thể viết dưới dạng đại lượng vectơ như sau:

$$Z = F(X)$$



Hình 2.22: Sơ đồ khối mạch tổ hợp

2. Phương pháp phân tích chức năng logic

Bài toán phân tích là bài toán từ sơ đồ logic cho trước, viết hàm logic của các đầu ra theo các đầu vào và nếu cần thì còn phải chỉ ra dạng sóng của tín hiệu ra tương ứng với tín hiệu vào. Xác định giá trị tín hiệu ở từng điểm trong sơ đồ. Thực hiện theo những bước sau:

2.1. Đặt các biến phụ

Tại đầu ra của mỗi cổng logic đặt một biến phụ.

2.2. Viết biểu thức

Tuần tự từ đầu vào đến đầu ra (hay cũng có thể ngược lại) viết ra biểu thức hàm logic của tín hiệu đầu ra theo các biến phụ, sau đó thay bằng các biến đầu vào của sơ đồ.

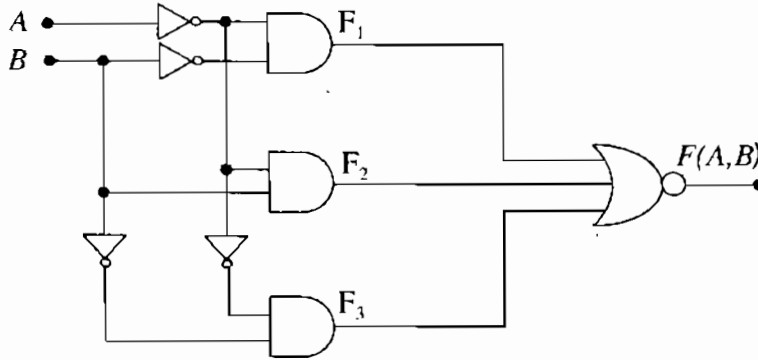
2.3. Rút gọn

Khi cần thiết thì rút gọn biểu thức trên đây bằng phương pháp đại số hay phương pháp hình vẽ.

2.4. Lập bảng logic

Khi cần thiết tìm ra bảng chân lý bằng cách tiến hành tính toán các giá trị hàm logic tín hiệu đầu ra tương ứng với tổ hợp có thể của các giá trị tín hiệu đầu vào.

*Ví dụ: Phân tích sơ đồ logic ở hình 2.23.



Hình 2.23: Phân tích mạch tổ hợp

Bước 1: Đặt các biến phụ F_1, F_2, F_3 như hình 2.23

Bước 2: Viết phương trình các biến phụ:

$$F = F_1 + F_2 + F_3$$

$$F_1 = \bar{A} B$$

$$F_2 = \bar{A} B$$

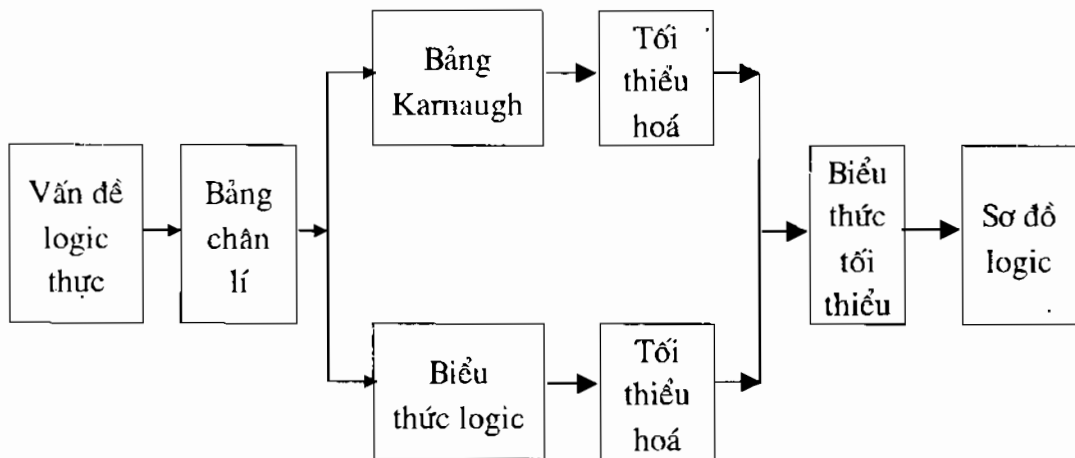
$$F_3 = \bar{A} B$$

Bước 3: Thay các giá trị của biến phụ vào hàm chính và rút gọn hàm (nếu hàm chưa rút gọn).

$$F = \bar{A} B + \bar{A} B + \bar{A} B$$

3. Phương pháp thiết kế logic mạch tổ hợp

Phương pháp thiết kế logic là các bước cơ bản tìm ra sơ đồ mạch điện logic từ yêu cầu nhiệm vụ logic đã cho.



Hình 2.24: Các bước thiết kế mạch logic tổ hợp

Hình 2.24 là quá trình thiết kế nói chung của mạch tổ hợp, trong đó bao gồm 4 bước chính:

3.1. Phân tích yêu cầu

Yêu cầu nhiệm vụ thiết kế của vấn đề logic thực có thể là một đoạn văn, cũng có thể là một bài toán logic cụ thể. Nhiệm vụ phân tích là xác định cái nào là biến số đầu vào, cái nào là hàm số đầu ra và mối quan hệ logic giữa chúng với nhau. Muốn phân tích đúng thì phải tìm hiểu xem xét một cách sâu sắc yêu cầu thiết kế. Việc đó là một việc khó nhưng quan trọng trong quá trình thiết kế.

3.2. Kê bảng chân lí

Nói chung, đầu tiên, chúng ta liệt kê thành bảng về quan hệ tương ứng nhau giữa trạng thái tín hiệu đầu vào với trạng thái hàm số đầu ra. Đó là bảng kê yêu cầu chức năng logic, gọi tắt là bảng chức năng. Việc này có vẻ dễ và trực quan. Tiếp theo, ta thay giá trị logic cho trạng thái, tức là dùng các số 0, số 1 biểu thị các trạng thái tương ứng của đầu vào và đầu ra. Kết quả ta có bảng các giá trị thực logic, gọi tắt là bảng chân lí. Đây chính là hình thức đại số của yêu cầu thiết kế.

3.3. Tiến hành tối thiểu hoá

Nếu số biến số tương đối ít thì có thể dùng phương pháp hình vẽ. Nếu số biến số tương đối nhiều, khi đó không tiện dùng phương pháp hình vẽ, thì dùng phương pháp đại số.

3.4. Vẽ sơ đồ logic

Kết quả việc tối thiểu hoá là biểu thức logic OR-AND. Căn cứ việc chọn loại cổng logic cụ thể, cần biến đổi biểu thức logic đó thành dạng phù hợp. Ví dụ, nếu chọn dùng cổng NAND thì phải có biểu thức dạng NAND, nếu chọn dùng cổng NOR thì phải có biểu thức dạng NOR, nếu chọn dùng cổng NORAND thì phải có biểu thức dạng NORAND.

Cần lưu ý rằng, những bước thiết kế trên đây không phải là bắt buộc áp dụng máy móc, mà là nên được vận dụng linh hoạt theo tình huống cụ thể của thiết kế thực tế.

Trong hoạt động thực tiễn của loài người, các bài toán và tương ứng các mạch điện của mạch logic tổ hợp là nhiều không kể hết. Các mạch tổ hợp hiện nay thường gặp trong hệ thống số là bộ mã hoá, bộ giải mã, bộ chọn kênh, bộ cộng, bộ kiểm tra chẵn lẻ (parity checker), ROM, v.v...

Ở những chương tiếp theo sẽ lần lượt giới thiệu một số mạch tổ hợp về cấu trúc mạch, nguyên lý công tác và đặc điểm của chúng, kết hợp soi sáng phương pháp thiết kế phân tích mạch tổ hợp.

3.5. Ví dụ về bài toán thiết kế mạch tổ hợp

Ví dụ: Một bóng đèn được điều khiển bằng 3 công tắc. Đèn chỉ sáng khi số lượng công tắc cùng đóng là quá bán. Hãy thiết kế mạch điều khiển trên.

Giải:

Bước 1: Phân tích nhiệm vụ

3 công tắc ký hiệu là K_1 , K_2 , và K_3 . Mỗi công tắc có 2 trạng thái *đóng* (nối mạch) và *mở* (hở mạch). Quy ước trạng thái đóng tương ứng với 1 logic, trạng thái mở tương ứng với 0 logic.

Đèn (ký hiệu là Đ) có hai trạng thái sáng và tắt. Chúng ta cũng quy ước: trạng thái đèn sáng tương ứng với 1 logic và trạng thái đèn tắt tương ứng với 0 logic.

Sự sáng tắt của đèn phụ thuộc vào tổ hợp đóng mở của các công tắc, vì vậy Đ là hàm và K biến số.

Bước 2: Lập bảng chân lý

Từ việc phân tích nhiệm vụ, ta lập được bảng chân lý 2.18:

Bảng 2.18

TT	Đầu vào			Đầu ra
	K_3	K_2	K_1	Đ
1	0	0	0	0
2	0	0	1	0
3	0	1	0	0
4	0	1	1	1
5	1	0	0	0
6	1	0	1	1
7	1	1	0	1
8	1	1	1	1

Bước 3: Viết hàm số và tối thiểu hàm

Dựa vào bảng chân lý vừa lập được, viết hàm số theo biểu thức logic (hoặc bảng Karnaugh). Trường hợp cụ thể ở đây chúng ta viết theo biểu thức logic theo dạng chuẩn tắc tuyển.

$$\bar{D} = \bar{K}_3 K_2 K_1 + K_3 \bar{K}_2 K_1 + K_3 K_2 \bar{K}_1 + K_3 K_2 K_1$$

Tối thiểu theo phương pháp đại số ta có:

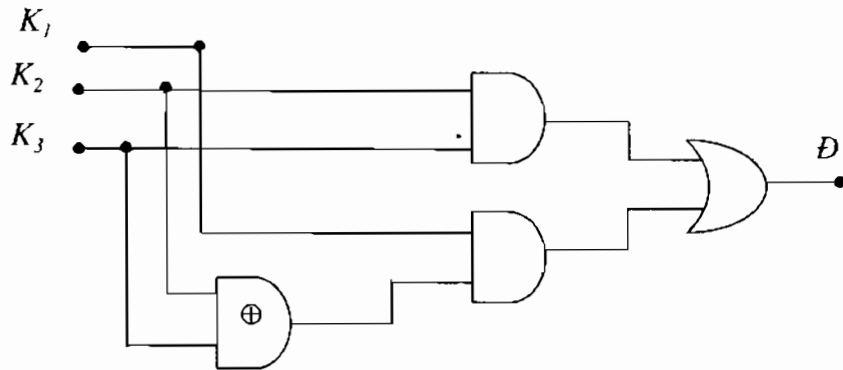
$$\bar{D} = K_1 (\bar{K}_3 K_2 + K_3 \bar{K}_2) + K_3 K_2 (\bar{K}_1 + K_1)$$

$$\bar{D} = K_1 (K_3 \oplus K_2) + K_3 K_2$$

Bước 4: Vẽ sơ đồ logic

Dựa vào hàm số vừa tối thiểu, vẽ sơ đồ logic.

Sơ đồ là hình 2.25.



Hình 2.25: Sơ đồ mạch điều khiển đèn

Câu hỏi

1. Trình bày khái niệm về biến và hàm logic.
2. Nêu các phép toán cơ bản trong đại số logic.
3. Trình bày về các cổng logic cơ bản:
 - a) Các khái niệm và các đặc tính tiêu biểu.
 - b) Sơ đồ ký hiệu các cổng và bảng chân lý của cổng.
 - c) Vẽ biểu đồ thời gian miêu tả hoạt động của các cổng logic.
4. Đặc điểm của mạch ba trạng thái.
5. Trình bày các phương pháp biểu diễn hàm logic.
6. Tối thiểu hoá các hàm logic sau bằng phương pháp đại số
 - a) $F = A \bar{B} C + A \bar{B} \bar{C}$
 - b) $F = A(BC + \bar{B} \bar{C}) + A(BC + \bar{B} \bar{C})$
 - c) $F = A \bar{B} + A \bar{B} CD(E + F)$
 - d) $F = AB + A \bar{C} + B \bar{C}$

7. Tối thiểu hoá các hàm logic sau bằng phương pháp bảng Karnaugh:

a) $F = \overline{A}B + A\overline{B} + AB$

b) $F = \overline{A}B\overline{C} + \overline{A}BC + A\overline{B}C + AB\overline{C}$

c) $F = \overline{A}BC + A\overline{B}C + ABC$

8. Cho hàm biểu diễn bằng bảng Karnaugh, (hình 2.26), tối thiểu hàm.

9. Trình bày về phương pháp phân tích và phương pháp thiết kế các mạch logic.

10. Phân tích các mạch tổ hợp sau, (hình 2.27) tối thiểu và vẽ lại sơ đồ theo hàm đã tối thiểu.

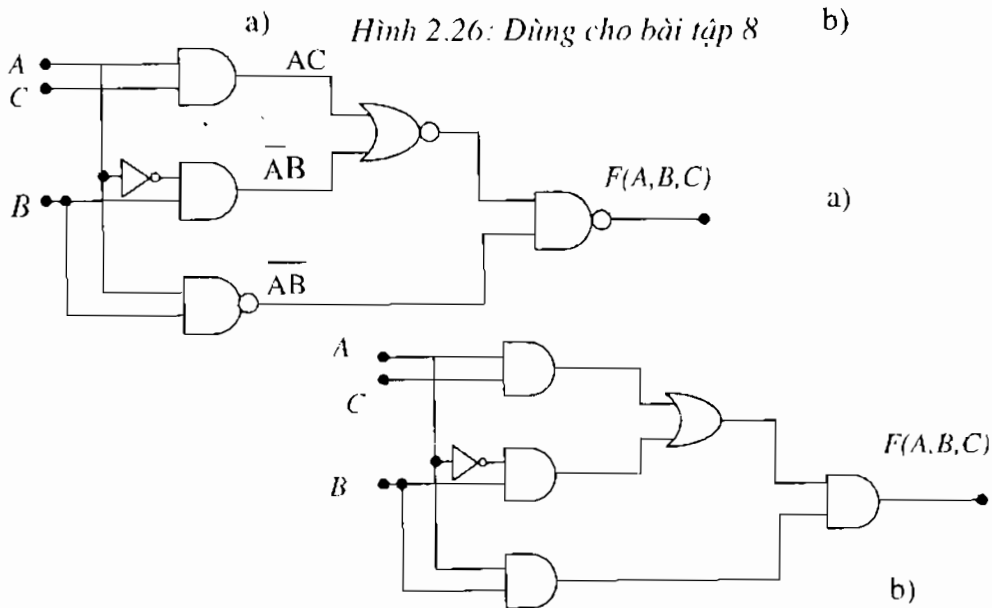
11. Thiết kế mạch tổ hợp sau:

Một động cơ hai chiều được điều khiển bằng bốn công tắc ký hiệu là A,B,C và D. Động cơ quay theo chiều thuận (ký hiệu là Y_1) khi 2 công tắc bất kỳ trong số 4 công tắc được đóng và động cơ quay theo chiều ngược lại (ký hiệu là Y_2) khi số công tắc được đóng là quá bán.

AB \ CD	00	01	11	10
00	1	1	1	
01			1	
11			1	
10	1			1

AB \ CD	00	01	11	10
00	1	1	1	1
01		1		
11	1			1
10				1

Hình 2.26: Dùng cho bài tập 8



Hình 2.27: Dùng cho bài tập 10

Chương 3

CÁC BỘ CỘNG, TRỪ VÀ SO SÁNH

Mục tiêu

- Phân tích được hoạt động của các bộ cộng, bộ trừ và bộ so sánh các số nhị phân.
- Rèn luyện được kỹ năng phân tích và thiết kế mạch tổ hợp nói chung.
- Hình thành được tác phong tư duy logic, tính cẩn thận chính xác.

Nội dung trọng tâm

- Nguyên lý làm việc của bộ cộng.
- Nguyên lý làm việc của bộ trừ.
- Nguyên lý làm việc của bộ so sánh.

I. BỘ CỘNG HAI SỐ NHỊ PHÂN

1. Bộ nửa tổng (Half Adder: HA)

1.1. Quy tắc thực hiện phép cộng

$$0 + 0 = 0$$

$$0 + 1 = 1$$

$$1 + 0 = 1$$

$$1 + 1 = 0 \text{ nhớ } 1$$

1.2. Thiết kế bộ nửa tổng

1.2.1. Lập bảng chân lý

Giả sử cộng hai số nhị phân A và B. Mỗi số có một bit. Kết quả phép cộng là S và số nhớ là C. Ta có bảng chân lý 3.1.

Đầu vào		Đầu ra	
A	B	S	C
0	0	0	0
0	1	1	0
1	0	1	0
1	1	0	1

1.2.2. Hàm số logic

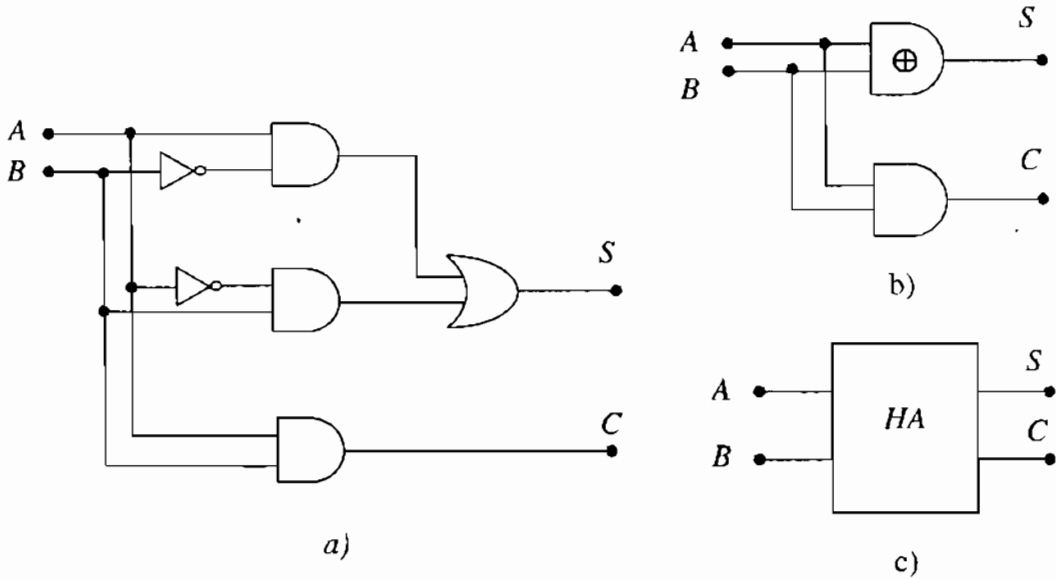
Từ bảng chân lý trên ta có các hàm số sau:

$$S = \overline{A}B + A\overline{B} = A \oplus B \quad (1)$$

$$C = A.B \quad (2)$$

1.2.3. Sơ đồ logic

Từ hàm số (1) và (2) ta có sơ đồ logic như hình 3.1.a và để gọn vẽ sơ đồ theo hình 3.1.b. Cuối cùng ký hiệu bộ nửa tổng theo hình 3.1.c.



Hình 3.1: Sơ đồ bộ nửa tổng

2. Bộ cộng đầy đủ (Full Adder : FA)

2.1. Khái quát chung

Trong thực tế ta phải thực hiện phép cộng những số nhị phân có nhiều bit, ví dụ:

$$\begin{array}{r} + \quad 1001 \\ \quad 1101 \\ \hline 10110 \end{array}$$

Ở ví dụ trên ta thấy ngay từ phép cộng 2 bit có trong số thấp nhất đã xuất hiện số nhớ. Số nhớ này được cộng vào cột có trong số cao hơn bên cạnh.

Ở bộ nửa tổng không có đầu vào cho số nhớ ở cột dưới đưa lên.

2.2. Thiết kế bộ cộng đầy đủ

Bộ cộng đầy đủ là bộ cộng, ngoài hai đầu vào của hai số hạng tham gia phép cộng còn có đầu vào để đưa số nhớ từ cột thấp lên.

2.2.1. Bảng chân lý

Để lập bảng chân lý ta có một số quy ước sau:

- Các số hạng tham gia phép cộng là A và B.
- Số nhớ từ cột dưới đưa lên là C_{n-1} .
- Tổng của phép cộng là S
- Số nhớ của phép cộng là C_n

Từ những quy ước trên ta có bảng chân lý 3.2 cho bộ cộng đầy đủ.

Bảng 3.2

Đầu vào			Đầu ra	
A	B	C_{n-1}	S	C_n
0	0	0	0	0
0	0	1	1	0
0	1	0	1	0
0	1	1	0	1
1	0	0	1	0
1	0	1	0	1
1	1	0	0	1
1	1	1	1	1

2.2.2. Các hàm số

Từ bảng chân lý của bộ cộng (bảng 3.2), viết hàm số S và C_n , sau đó tối thiểu các hàm số đó.

$$S = \overline{A}\overline{B}C_{n-1} + \overline{A}B\overline{C}_{n-1} + A\overline{B}\overline{C}_{n-1} + ABC_{n-1}$$

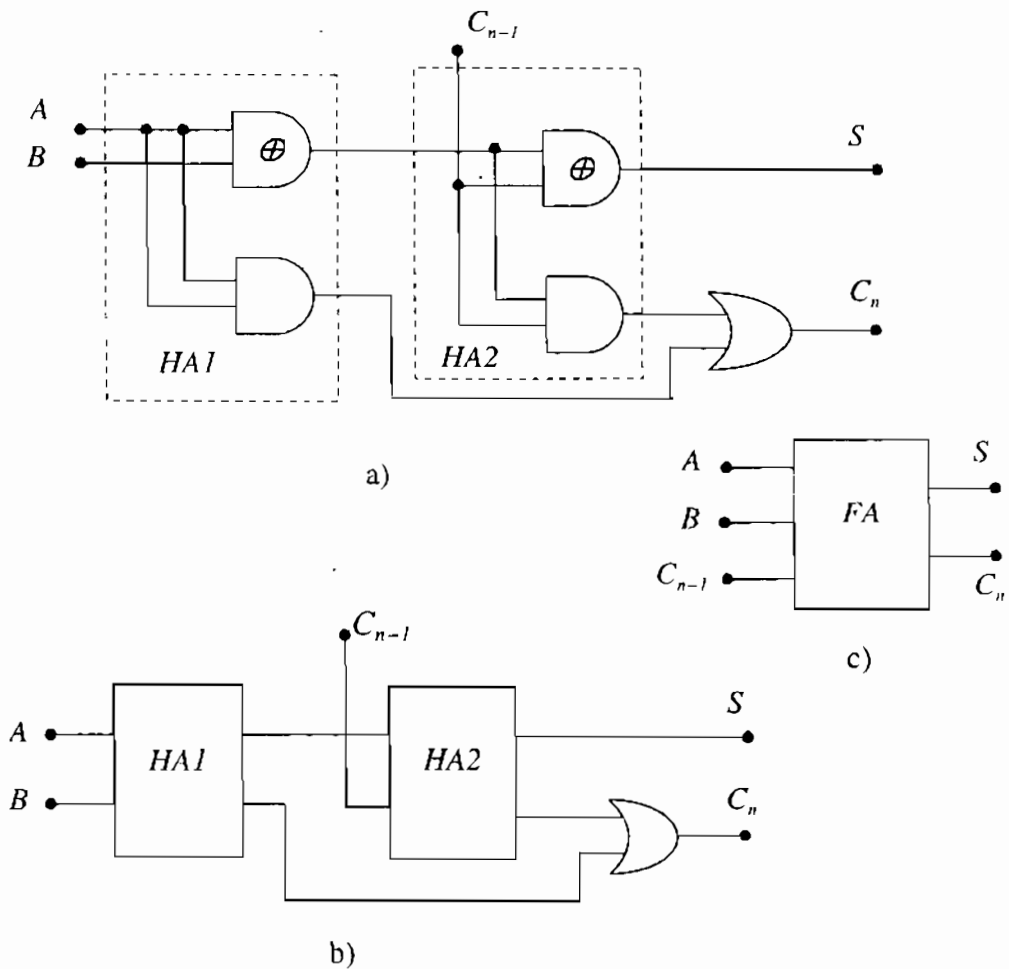
$$\text{Tối thiểu hàm trên ta có } S = C_{n-1} \oplus (A \oplus B) \quad (1)$$

$$C_n = \overline{A}B C_{n-1} + A\overline{B} C_{n-1} + A B \overline{C}_{n-1} + A B C_{n-1}$$

$$\text{Tối thiểu } C_n \text{ ta có } C_n = C_{n-1} (A \oplus B) + AB \quad (2)$$

2.2.3. Sơ đồ bộ cộng đầy đủ

Từ hàm số (1) và (2) ở mục 2.2.2 ta có sơ đồ như hình 3.2.a. Từ 3.2.a ta thấy sơ đồ gồm hai bộ HA và một cổng OR. Ta vẽ lại được hình 3.2.b. Để sơ đồ được gọn hơn ta ký hiệu bộ cộng đầy đủ như hình 3.2.c.



Hình 3.2: Sơ đồ bộ cộng dây chuyền

3. Bộ cộng song song 4 bit

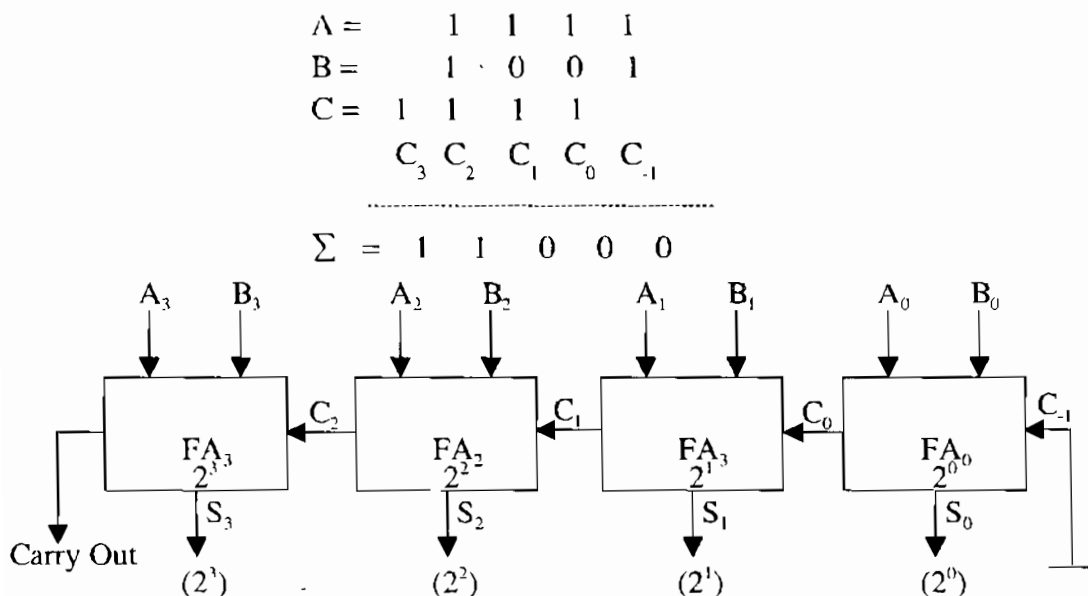
Giả sử ta có hai số nhị phân 4 bit A và B, trong đó:

$$A = A_3 A_2 A_1 A_0$$

$$B = B_3 B_2 B_1 B_0$$

Với A_0, B_0 là cột có trọng số bé nhất của hai số A và B (2^0), A_3, B_3 là cột có trọng số lớn nhất của hai số A và B (2^3). Từ bộ cộng nhị phân 1 bit ta có thể đưa ra mạch của bộ cộng song song 4 bit thực hiện phép cộng hai số A, B như hình 3.3.

Ví dụ: Có hai số A và B với A = 1111, B = 1001. Phép cộng hai số này được thực hiện như sau.



Hình 3.3: Bộ cộng song song 4 bit

Nhược điểm của bộ cộng loại này:

Tín hiệu nhớ ở đầu ra (Carry out) được tạo thành sau khi đi qua bốn bộ cộng $FA_0 \dots FA_3$, nếu mỗi bộ cộng FA_i là mạch hai tầng thì Carry out sẽ xuất hiện ở đầu ra sau thời gian $T = 8$ lần thời gian quá độ, với thời gian quá độ là thời gian quá độ của một tầng logic. Do vậy, thời gian trễ sẽ rất lớn, đặc biệt khi số bit của mỗi số hạng tăng đủ lớn. Để khắc phục nhược điểm đó, người ta dùng bộ cộng nhớ nhanh hay là bộ cộng nhớ nhìn trước (Fast Carry hay Carry Look Ahead).

* Bộ cộng nhớ nhanh (Fast Carry hay Carry Look Ahead)

Như đã trình bày ở trên, trong bộ cộng song song n bit số nhớ truyền liên tiếp từ tầng này sang tầng khác làm cho thời gian thực hiện phép cộng lớn, đặc biệt là khi n lớn. Do vậy, cần có biện pháp khắc phục nhược điểm đó. Một trong những phương pháp đã được áp dụng là phương pháp nhớ nhanh (Fast Carry) hay là nhớ nhìn trước (Carry Look Ahead) sẽ được trình bày ở đây. Đối với một bộ cộng đầy đủ (FA) một cột số, ta có:

$$C_i = (A_i \oplus B_i) C_{i-1} + A_i B_i \quad (3-1)$$

$$S_i = A_i \oplus B_i \oplus C_{i-1}$$

$$\text{đặt: } A_i \oplus B_i = P_i$$

$$A_i B_i = G_i \quad (3-2)$$

Thay (3-1) vào (3-2) ta có:

$$C_i = P_i C_{i-1} + G_i$$

$$S_i = P_i \oplus C_{i-1}$$

Đối với bộ cộng 4 bit ta có:

$$C_0 = P_0 C_{-1} + G_0$$

$$C_1 = P_1 C_0 + G_1$$

$$C_2 = P_2 C_1 + G_2$$

$$C_3 = P_3 C_2 + G_3$$

Thay C_0 vào C_1 , C_1 vào C_2 , và C_2 vào C_3 , ta được kết quả:

$$C_1 = P_1 P_0 C_{-1} + P_1 G_0 + G_1$$

$$\left\{ \begin{array}{l} C_2 = P_2 P_1 P_0 C_{-1} + P_2 P_1 G_0 + P_2 G_1 + G_2 \end{array} \right. \quad (3-3)$$

$$\left\{ \begin{array}{l} C_3 = P_3 P_2 P_1 P_0 C_{-1} + P_3 P_2 P_1 G_0 + P_3 P_2 G_1 + P_3 G_2 + G_3 \end{array} \right.$$

Các chữ số của tổng sẽ có giá trị như sau:

$$S_0 = P_0 \oplus C_{-1}$$

$$S_1 = P_1 \oplus C_0$$

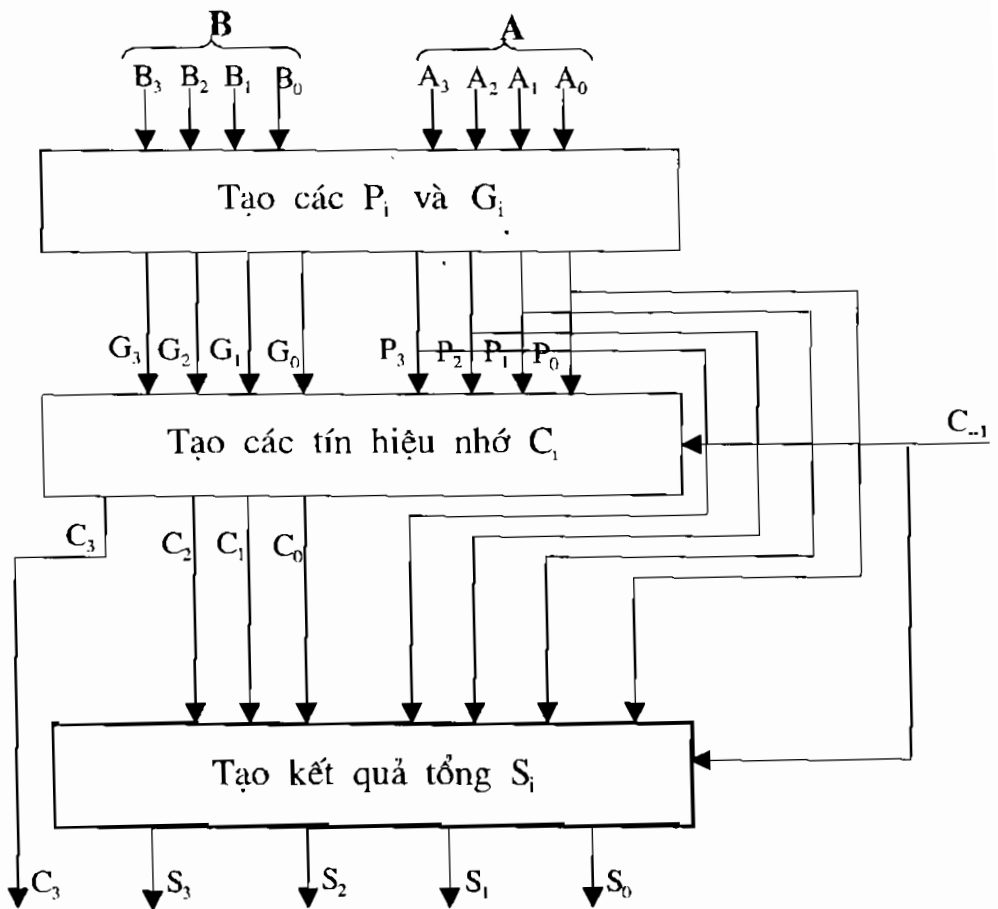
$$S_2 = P_2 \oplus C_1 \quad (3-4)$$

$$S_3 = P_3 \oplus C_2$$

Dùng hệ phương trình (3-3) và (3-4) để xây dựng bộ cộng song song nhớ nhanh. Dễ dàng nhận thấy rằng bộ cộng này gồm 3 khối chính, đó là:

- Khối tạo các giá trị P_i và G_i
- Khối tạo các tín hiệu nhớ C_i
- Khối tạo kết quả phép cộng S_i

Sơ đồ khối của bộ cộng song song 4 bit nhớ nhanh được mô tả trong hình 3. 4.



Hình 3. 4: Sơ đồ khối bộ cộng song song 4 bit nhớ nhanh

II. BỘ TRỪ HAI SỐ NHỊ PHÂN

1. Bộ nửa trừ (Half Subtractor: HS)

1.1. Quy tắc thực hiện phép trừ

$$0 - 0 = 0$$

$10 - 1 = 1$ vay 1 từ cột có trọng số cao hơn liền kề.

$$1 - 0 = 1$$

$$1 - 1 = 0$$

1.2. Thiết kế bộ nửa trừ

1.2.1. Lập bảng chân lý

Để lập bảng chân lý bộ nửa trừ, (bảng 3. 3) ta có một số quy ước như sau:

Gọi số bị trừ là A, số trừ là B, hiệu số là D và số nhớ là C.

Bảng 3.3

Đầu vào		Đầu ra	
A	B	D	C
0	0	0	0
0	1	1	1
1	0	1	0
1	1	0	0

1.2.2. Hàm số logic

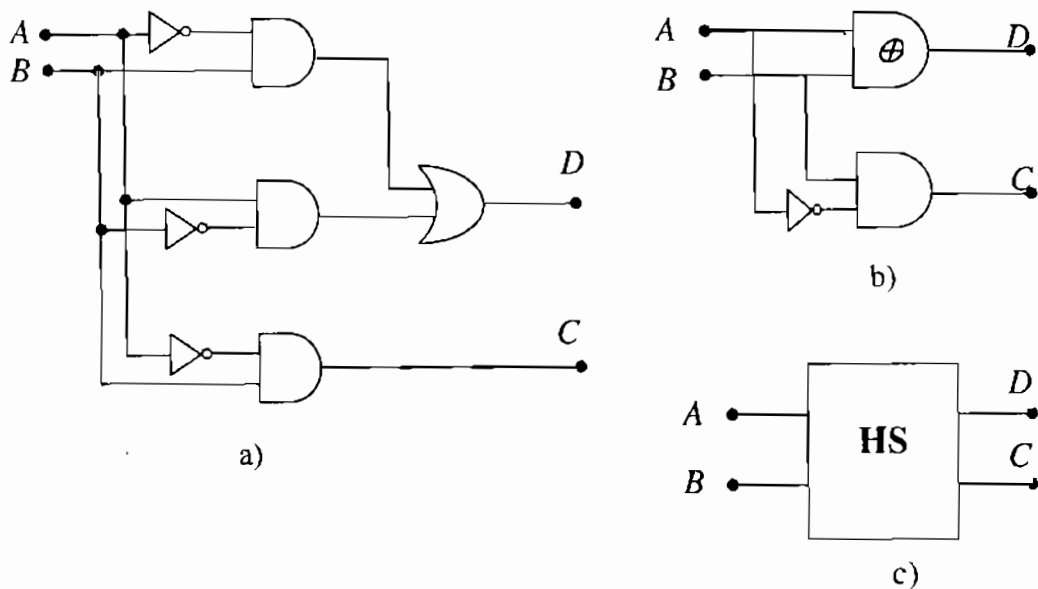
Từ bảng chân lý của bộ nửa trừ ta có các hàm số D và C viết theo dạng chuẩn tắc tuyển:

$$D = \bar{A}B + A\bar{B} = A \oplus B$$

$$C = \bar{A}B$$

1.2.3. Sơ đồ bộ nửa trừ

Dựa vào các hàm số của D và C ở mục 1.2.2 ta có sơ đồ bộ nửa trừ theo hình 3.5.



Hình 3.5: Sơ đồ bộ nửa trừ

Từ hình 3.5.a. vẽ gọn lại được hình 3.5.b và ký hiệu cho gọn hơn bằng hình 3.5.c.

2. Bộ trừ đầy đủ (Full Subtractor: FS)

2.1. Khái quát

Cũng như ở bộ cộng đầy đủ, cần phải có đầu vào cho số nhớ từ cột có trọng số thấp đưa lên. Ở bộ trừ cũng như vậy. Qua ví dụ sau ta thấy rõ điều này.

Ví dụ:

$$\begin{array}{r} 1100 \\ - 1001 \\ \hline 0011 \end{array}$$

$$\begin{array}{r} 1001 \\ - 1101 \\ \hline 0011 \end{array}$$

Khi thực hiện phép trừ 0-1 ta phải mượn 1 từ cột có trọng số cao hơn liền kề. Sau đó khi thực hiện phép trừ ở cột có trọng số cao hơn liền kề ta phải trả cho cột đó. Cách trả cũng tương tự như hệ mười.

Như vậy, để có bộ trừ đầy đủ ngoài số bị trừ là A, số trừ là B, đầu vào của bộ trừ đầy đủ cần có thêm đầu vào C_{n-1} là đầu đưa số nhớ từ cột có trọng số thấp lên. Chúng ta tiến hành thiết kế bộ trừ đầy đủ như sau:

2.2. Thiết kế bộ trừ đầy đủ

2.2.1. Lập bảng chân lý

Một số quy ước

- Ký hiệu A: số bị trừ.
- Ký hiệu B: số trừ.
- Ký hiệu C_{n-1} : số nhớ từ cột dưới đưa lên.

- Ký hiệu S: hiệu số.
- Ký hiệu C_n : số nhớ của cột n.

Dựa vào quy tắc thực hiện phép trừ và các quy ước trên ta lập được bảng chân lý cho bộ trừ đầy đủ như bảng 3.4.

Bảng 3.4

Đầu vào			Đầu ra	
A	B	C_{n-1}	D	C_n
0	0	0	0	0
0	0	1	1	1
0	1	0	1	1
0	1	1	0	1
1	0	0	1	0
1	0	1	0	0
1	1	0	0	0
1	1	1	1	1

2.2.2. Hàm số logic

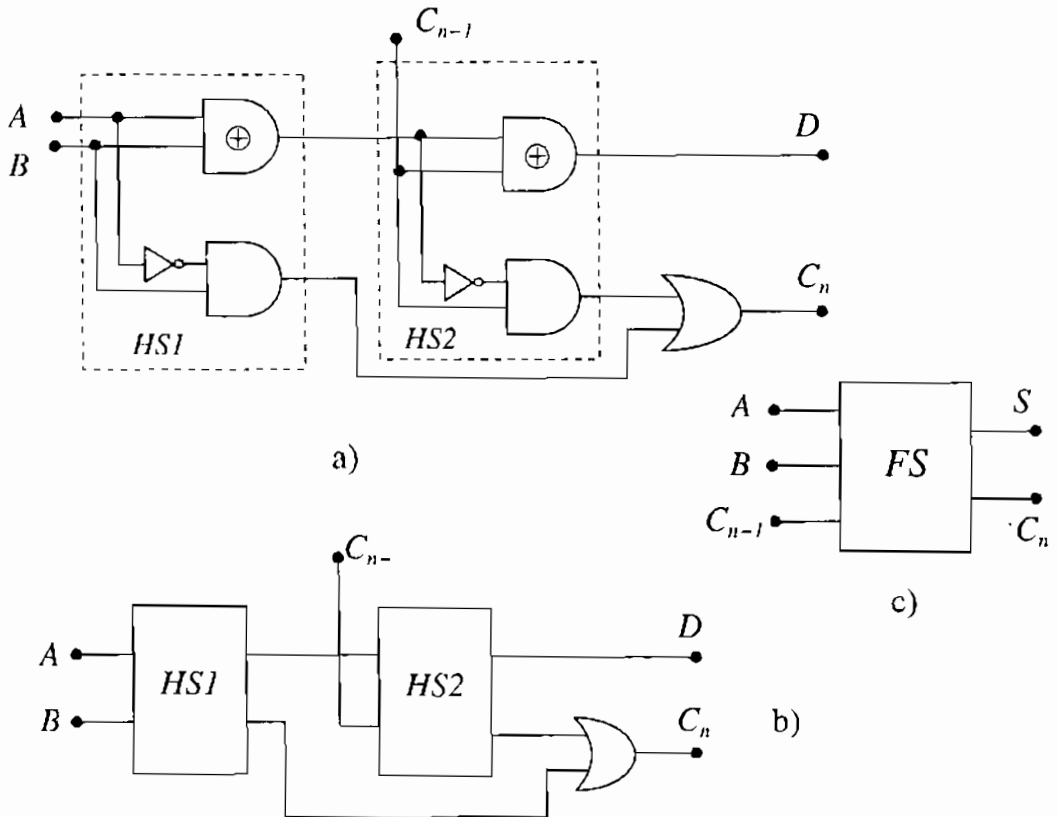
Từ bảng chân lý của bộ trừ đầy đủ, ta viết các hàm số D và C_n theo dạng chuẩn tắc tuyển rồi tiến hành tối thiểu hoá hàm số.

$$\begin{aligned} D &= \overline{A}\overline{B}C_{n-1} + \overline{A}B\overline{C}_{n-1} + A\overline{B}\overline{C}_{n-1} + ABC \\ &= C_{n-1}(\overline{A}\overline{B} + AB) = C_{n-1}(\overline{A}\overline{B} + AB)\overline{C}_{n-1} \\ &= (\overline{A} \oplus B)C_{n-1} + (\overline{A} \oplus B)\overline{C}_{n-1} \\ &= (\overline{A} \oplus B)C_{n-1} + \overline{A}B\overline{C}_{n-1} \end{aligned} \quad (1)$$

$$\begin{aligned} S &= \overline{A}\overline{B}C_{n-1} + \overline{A}B\overline{C}_{n-1} + A\overline{B}\overline{C}_{n-1} + C_{n-1} \\ &= (\overline{A}\overline{B} + AB)C_{n-1} + \overline{A}B(\overline{C}_{n-1} + C_{n-1}) \\ &= (\overline{A} \oplus B)C_{n-1} + \overline{A}B \end{aligned} \quad (2)$$

2.2.3. Sơ đồ bộ trừ đầy đủ

Từ (1) và (2) vẽ sơ đồ 3.6.a, sơ đồ gồm 2 bộ HS. Để gọn ta vẽ lại như hình 3.6.b. Cuối cùng ký hiệu bộ trừ đầy đủ như hình 3.6.c.



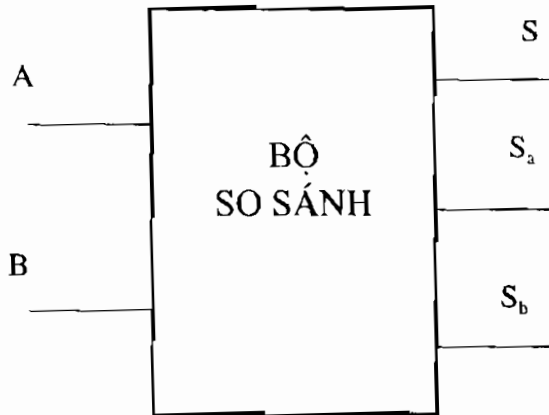
Hình 3.6: Sơ đồ bộ trừ đầy đủ

III. BỘ SO SÁNH HAI SỐ NHỊ PHÂN

Bộ so sánh số dùng để so sánh định tính hai số nhị phân. Kết quả so sánh được thể hiện bằng các giá trị logic ở đầu ra chỉ sự bằng nhau, lớn hơn hay bé hơn của hai số nhị phân.

1. Bộ so sánh hai số nhị phân một bit

Bộ so sánh này thực hiện việc so sánh hai số nhị phân, mỗi số có độ dài là 1 bit. Hai số nhị phân cần so sánh là A và B. Khi thực hiện so sánh, nếu kết quả so sánh $A = B$ đầu ra $S = 1$ logic, nếu $A > B$ đầu ra $S_a = 1$ logic còn nếu $A < B$ thì đầu ra $S_b = 1$. Sơ đồ khối của bộ so sánh ký hiệu như hình 3.7.



Hình 3.7: Sơ đồ khối bộ so sánh

Từ phân tích trên ta có bảng chân lý miêu tả hoạt động của bộ so sánh hai số nhị phân một bit ở bảng 3.5:

Bảng 3.5

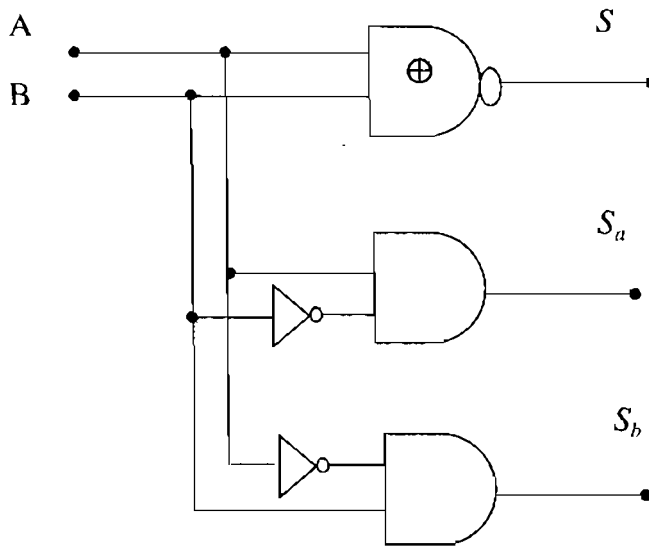
Đầu vào		Đầu ra		
A	B	S	S_a	S_b
0	0	1	0	0
0	1	0	0	1
1	0	0	1	0
1	1	1	0	0

Từ bảng chân lý 3.5 ta viết được các hàm logic như sau:

$$S = \overline{A} \overline{B} + A B = \overline{A \oplus B}$$

$$S_a = \overline{A} B$$

$$S_b = A \overline{B}$$



Hình 3.8: Sơ đồ bộ so sánh hai số 1 bit

2. Bộ so sánh hai số nhị phân hai bit

Giả sử cần so sánh hai số nhị phân, mỗi số có độ dài là 2 bit.

Số thứ nhất $A = a_1a_0$

Số thứ hai $B = b_1b_0$.

Ở đây ta cũng quy ước khi $A = B$ có $S = 1$, khi $A > B$ có $S_a = 1$ và khi $A < B$ có $S_b = 1$.

2.1. Trường hợp $A = B$

Khi $A = B$, tức là $S = 1$, ta có bảng chân lý sau (bảng 3.6)

Bảng 3.6

A		B		S
A_1	A_0	B_1	B_0	
0	0	0	0	1
0	1	0	1	1
1	0	1	0	1
1	1	1	1	1

Từ bảng chân lý trên ta viết hàm số cho S theo dạng chuẩn tắc tuyển:

$$S = \overline{A_1} \overline{A_0} \overline{B_1} \overline{B_0} + \overline{A_1} A_0 \overline{B_1} B_0 + A_1 \overline{A_0} \overline{B_1} \overline{B_0} + A_1 A_0 B_1 B_0$$

Tối thiểu hàm số theo phương pháp đại số ta có:

$$\begin{aligned} S &= \overline{A_1} \overline{B_1} (\overline{A_0} \overline{B_0} + A_0 B_0) + A_1 \overline{B_1} (\overline{A_0} \overline{B_0} + A_0 B_0) \\ &= \overline{A_1} \overline{B_1} (A_0 \oplus B_0) + A_1 \overline{B_1} (A_0 \oplus B_0) \\ &= (A_0 + B_0) + (A_1 + B_1) \end{aligned}$$

2.2. Trường hợp A > B

Ở trường hợp này ta có $S_a = 1$. Bảng 3.7 là bảng chân lý cho trường hợp này.

Bảng 3.7

A		B		S_a
A_1	A_0	B_1	B_0	
0	1	0	0	1
1	0	0	0	1
1	0	0	1	1
1	1	0	0	1
1	1	0	1	1
1	1	1	0	1

Dựa vào bảng chân lý ta có hàm S_a như sau:

$$\begin{aligned} S_a &= \overline{A_1} A_0 \overline{B_1} \overline{B_0} + A_1 \overline{A_0} \overline{B_1} \overline{B_0} + A_1 \overline{A_0} \overline{B_1} B_0 + A_1 A_0 \overline{B_1} \overline{B_0} + A_1 A_0 \overline{B_1} B_0 \\ &\quad + A_1 A_0 B_1 \overline{B_0} \end{aligned}$$

Tối thiểu hoá hàm số ta có:

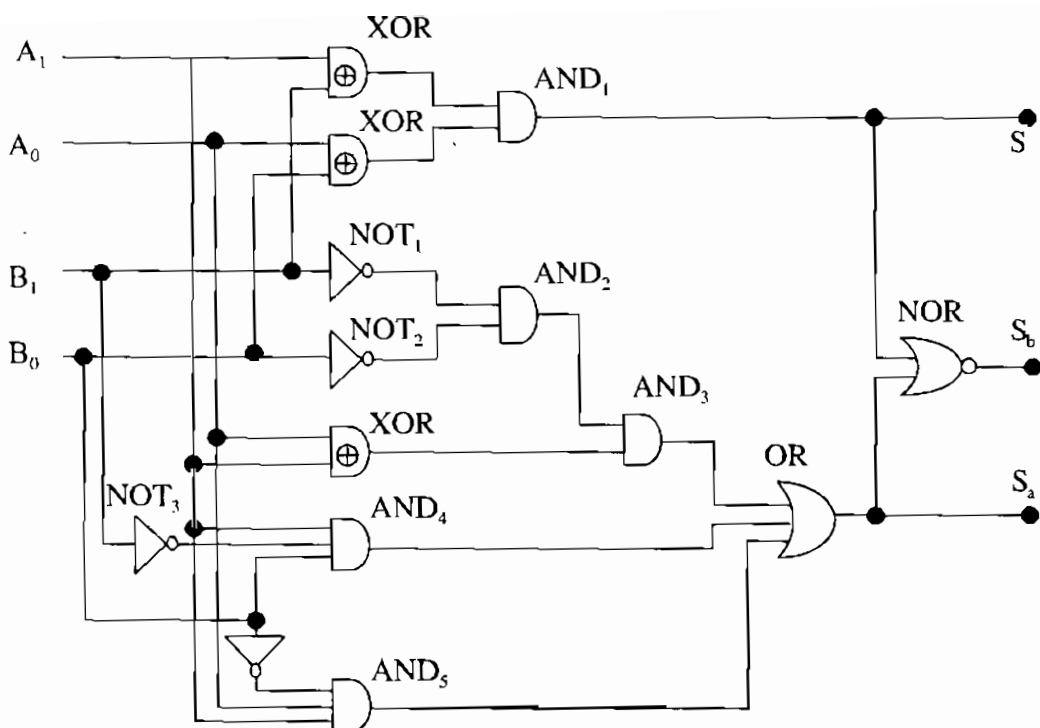
$$\begin{aligned} S_a &= B_1 B_0 (A_1 A_0 + A_1 \overline{A_0}) + A_1 \overline{B_1} B_0 (A_0 + \overline{A_0}) + A_1 A_0 B_0 (B_1 + \overline{B_1}) \\ &= B_1 B_0 (A_1 \oplus A_0) + A_1 \overline{B_1} B_0 + A_1 A_0 B_0 \\ &= B_1 B_0 (A_1 \oplus A_0) + A_1 (B_1 B_0 + A_0 B_0) \end{aligned}$$

2.3. Trường hợp A < B ($S_b = 1$)

$A < B$ khi không phải $A = B$ và không phải $A > B$, từ đây suy ra $S_b = \overline{S} \cdot \overline{S_a}$ hay bằng $\overline{S + S_a}$

2.4. Sơ đồ logic

Dựa vào các hàm số đã tối thiểu ở các mục 2.1, 2.2 và 2.3 ta có sơ đồ bộ so sánh 2 số nhị phân như hình 3.9.



Hình 3.9: Bộ so sánh 2 số 2 bit

3. Trường hợp tổng quát

Giả sử có 2 số nhị phân n bit A, B với:

$$A = A_n A_{n-1} \dots A_1$$

$$B = B_n B_{n-1} \dots B_1$$

Trong đó quy ước A₁, B₁ là cột có trọng số nhỏ nhất, A_n, B_n là cột có trọng số lớn nhất. Để xây dựng sơ đồ mạch bộ so sánh này có thể dùng 2 cách như sau:

3.1. Xây dựng sơ đồ trực tiếp từ các hàm

A > B (2n biến)

A = B (2n biến)

A < B (2n biến).

Với phương pháp này, thực chất là xây dựng một hệ 3 hàm logic, mỗi hàm có 2n biến.

3.2. Xây dựng sơ đồ gián tiếp qua các bộ so sánh 1 bit

Xét cụ thể với n = 3

$$A = A_3 A_2 A_1$$

$$B = B_3 B_2 B_1$$

Để xây dựng được sơ đồ mạch so sánh này, phải cần 3 mạch so sánh 1 bit và các mạch phụ trợ.

Đầu tiên so sánh 2 cột số có trọng số lớn nhất A_3, B_3 :

$$A_3 > B_3: A > B$$

$$A_3 < B_3: A < B$$

$$A_3 = B_3: \text{So sánh tiếp } A_2, A_1 \text{ với } B_2, B_1$$

Trong trường hợp $A_3 = B_3$, ta phải tiếp tục so sánh 2 chữ số A_2, B_2 có thể xảy ra trường hợp:

$$A_2 > B_2: A > B$$

$$A_2 < B_2: A < B$$

$$A_2 = B_2: \text{so sánh tiếp}$$

Nếu $A_2 = B_2$, ta phải tiếp tục so sánh 2 chữ số A_1, B_1 :

$$A_1 > B_1: A > B$$

$$A_1 < B_1: A < B$$

$$A_1 = B_1: A = B$$

Tóm lại:

$$(A = B) \Leftrightarrow (A_3 = B_3) (A_2 = B_2) (A_1 = B_1)$$

$$(A > B) \Leftrightarrow (A_3 > B_3) + (A_3 = B_3) (A_2 > B_2) + (A_3 = B_3) (A_2 = B_2) (A_1 > B_1)$$

$$(A < B) \Leftrightarrow (A_3 < B_3) + (A_3 = B_3) (A_2 < B_2) + (A_3 = B_3) (A_2 = B_2) (A_1 < B_1)$$

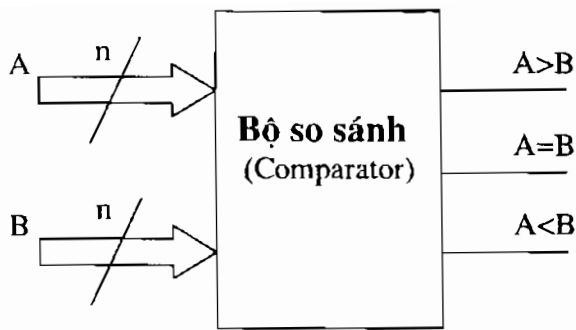
Trên cơ sở đó xây dựng được sơ đồ bộ so sánh 2 số 3 bit dùng các bộ so sánh 1 bit và các mạch NAND như hình 3.10.b. Trong đó:

$$A_3 = B_3, A_3 > B_3, A_3 < B_3$$

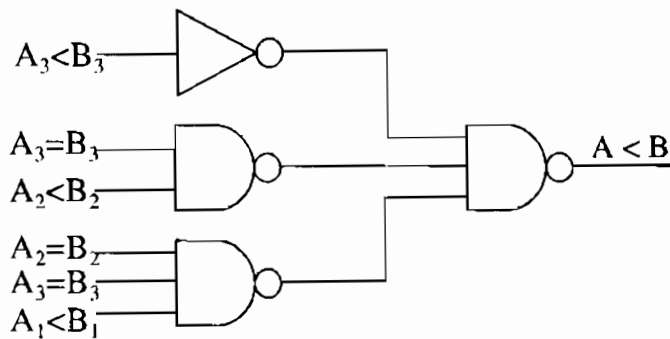
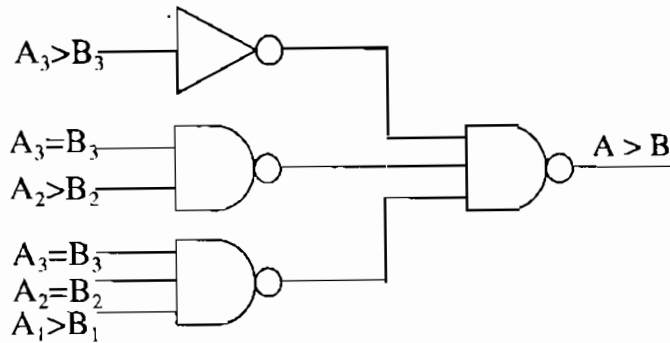
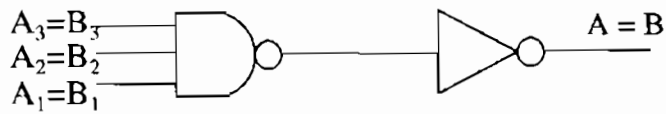
$$A_2 = B_2, A_2 > B_2, A_2 < B_2$$

$$A_1 = B_1, A_1 > B_1, A_1 < B_1$$

được đưa từ các đầu ra của 3 bộ so sánh 1 bit đến.



a)



b)

Hình 3.10: Bộ so sánh nhiều bit

a- Sơ đồ khối của bộ so sánh bit.

b- Sơ đồ của bộ so sánh 3 bit được xây dựng từ bộ so sánh 1 bit.

Câu hỏi và bài tập

1. Xây dựng bộ cộng hai số nhị phân đầy đủ.
2. Vẽ biểu đồ mô tả tổng S và số nhớ C_n khi thực hiện cộng hai số hạng A, B và số nhớ C_{n-1} như sau:
A = 1 1 1 1 0 0 1 0
B = 1 0 1 1 0 1 1 1
 $C_{n-1} = 1 0 0 1 1 1 1 0$
3. Xây dựng bộ trừ hai số nhị phân đầy đủ.
4. Vẽ biểu đồ mô tả hiệu số D và số nhớ C_n khi thực hiện A trừ B và số nhớ C_{n-1} như sau:
A = 1 1 1 1 0 0 1 0
B = 1 0 1 1 0 1 1 1
 $C_{n-1} = 1 0 0 1 1 1 1 0$
5. Phân tích hoạt động của bộ so sánh hai số nhị phân 2 bit.

Chương 4

BỘ DẪN KÊNH VÀ BỘ PHÂN KÊNH

Mục tiêu

- Phân tích được nguyên lý làm việc của các bộ dẫn kênh và phân kênh.
- Biết ứng dụng các mạch dẫn kênh và phân kênh để tạo được một số mạch thực hiện các chức năng khác.
- Rèn luyện khả năng phân tích và tư duy logic trong chuyên ngành học.

Nội dung trọng tâm

- Phân tích nguyên lý làm việc của bộ dẫn kênh và phân kênh.
- Ứng dụng bộ dẫn kênh và phân kênh.

I. KHÁI QUÁT CHUNG

Bình thường, để truyền thông tin song song n bit cần có n đường dây để truyền. Nhưng nếu như vậy mạch điện sẽ rất phức tạp. Để khắc phục điều này người ta đưa ra phương pháp tập trung thông tin lên một đường dây để truyền và ở đầu cuối, thông tin lại được tách ra theo nguyên tắc thể hiện qua hình vẽ 4.1 dưới đây.

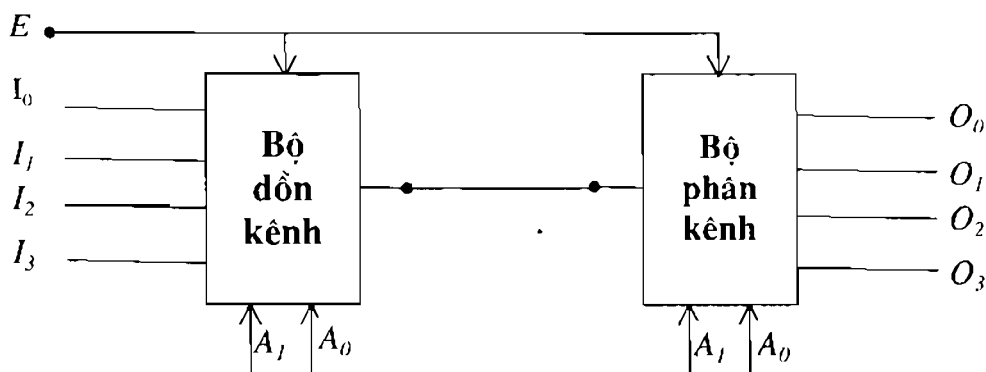


Hình 4.1: Sơ đồ nguyên tắc truyền tin song song

Chuyển mạch A sẽ lựa chọn từng đầu vào để thông tin ở từng đầu vào đó được truyền trên một đường dây chung. Chuyển mạch B có nhiệm vụ chọn đầu

ra để đưa thông tin ra đúng tuyến tương ứng với đầu vào của thông tin cần truyền. Để thông tin đi đúng tuyến, ở chuyển mạch A và B cần có tín hiệu điều khiển C để chọn từng đầu vào cũng như việc chọn từng đầu ra.

Trong kỹ thuật điện tử số, các chuyển mạch A và B được xây dựng bằng các mạch logic tổ hợp. Chuyển mạch A được gọi là bộ dồn kênh (hoặc gọi là bộ chọn kênh - Multiplexer: MUX). Chuyển mạch B được gọi là bộ phân kênh (hoặc gọi là bộ chia kênh- Demultiplexer: DEMUX). Hình 4.2 là sơ đồ truyền thông tin song song dùng bộ dồn kênh và phân kênh. Ở sơ đồ này có bốn đầu vào dữ liệu ($I_0 \rightarrow I_3$) và bốn đầu ra dữ liệu từ ($O_0 \rightarrow O_3$).



Hình 4.2: Truyền tin song song bằng bộ dồn kênh và phân kênh

Với bốn đầu vào và bốn đầu ra dữ liệu như vậy nên ở bộ dồn kênh cũng như bộ phân kênh đều cần hai đầu vào điều khiển, được ký hiệu là A_0 và A_1 . Hai đầu vào điều khiển giúp cho việc chọn một trong bốn đầu vào (đối với bộ dồn kênh) và một trong bốn đầu ra (đối với bộ phân kênh). Đối với bộ dồn kênh trong trường hợp cụ thể này, việc chọn đầu để đưa dữ liệu vào nhờ tín hiệu đưa vào A_0 và A_1 được thực hiện như sau:

- Khi trạng thái logic của $A_1, A_0 = 00$ chọn đầu vào I_0 .
- Khi trạng thái logic của $A_1, A_0 = 01$ chọn đầu vào I_1 .
- Khi trạng thái logic của $A_1, A_0 = 10$ chọn đầu vào I_2 .
- Khi trạng thái logic của $A_1, A_0 = 11$ chọn đầu vào I_3 .

Đối với bộ phân kênh trong trường hợp này cũng tương tự.

- Khi trạng thái logic của $A_1, A_0 = 00$ chọn đầu ra O_0 .
- Khi trạng thái logic của $A_1, A_0 = 01$ chọn đầu ra O_1 .

- Khi trạng thái logic của $A_1, A_0 = 10$ chọn đầu ra O_2 .
- Khi trạng thái logic của $A_1, A_0 = 11$ chọn đầu ra O_3 .

Như vậy, ta thấy giữa các đầu vào (ra) dữ liệu và các đầu vào điều khiển (A_i) để chọn đầu dữ liệu vào, ra có mối liên quan chặt chẽ với nhau. Số đầu vào điều khiển là n thì sẽ điều khiển chọn (quản lý) được 2^n đầu vào, (ra) dữ liệu.

Ở sơ đồ hình 4.2, đầu vào E (Enable) là đầu vào cho phép bộ dồn kênh và phân kênh hoạt động. Tùy theo trạng thái tích cực của E đối với bộ dồn kênh và phân kênh cụ thể mà nó cho phép bộ dồn kênh cũng như bộ phân kênh được làm việc hay bị khoá lại.

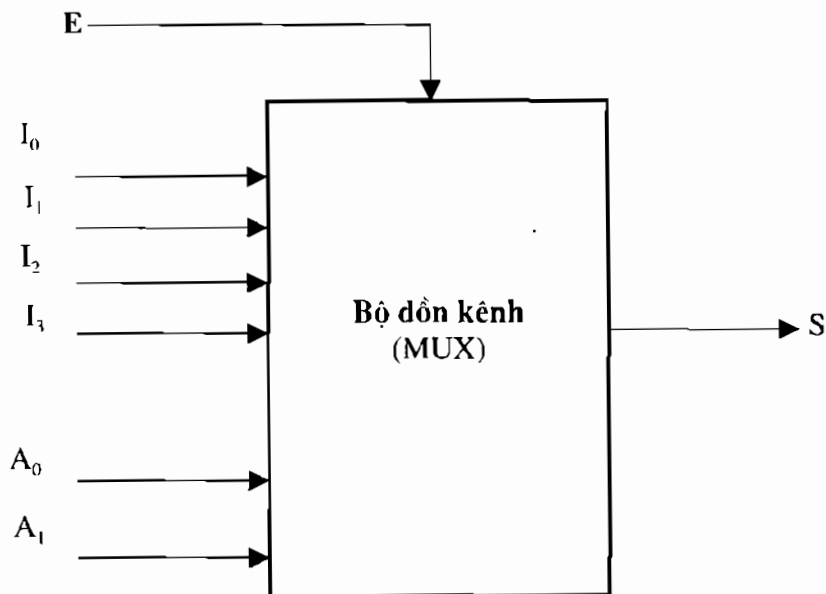
II. BỘ DỒN KÊNH

1. Nhiệm vụ

Biến đổi thông tin vào song song thành thông tin ra nối tiếp.

Dưới sự điều khiển của n tín hiệu chọn, thực hiện sự chọn ra một đầu vào (một kênh) trong số 2^n đầu vào để đưa tín hiệu ở đầu vào được chọn tới đầu ra. (Có một đầu ra).

2. Sơ đồ bộ dồn kênh 4 bit



Hình 4.3: Sơ đồ khối bộ dồn kênh 4 bit

Hình 4.3 vẽ sơ đồ khối của một bộ dồn kênh 4 bit (4 đầu vào dữ liệu). Với bốn đầu vào dữ liệu là I_0, I_1, I_2, I_3 . Để chọn được từng đầu vào riêng biệt dùng 2 đường địa chỉ điều khiển chọn là A_0 và A_1 . Giả sử bộ dồn kênh được làm việc khi $E = "1"$ và bị khoá khi $E = "0"$. Bảng chân lý 4.1 mô tả hoạt động của bộ dồn kênh 4 bit.

- Khi $E = 0$, dù cho A_0, A_1 ở trạng thái nào thì cũng không chọn được một đầu vào dữ liệu nào của bộ dồn kênh. Đầu ra của bộ dồn kênh vì vậy bằng 0.

- Khi $E = 1$, ứng với mỗi trạng thái của A_0 và A_1 , một đầu vào dữ liệu được chọn và dữ liệu ở đầu được chọn đó được đưa ra đầu ra của bộ dồn kênh.

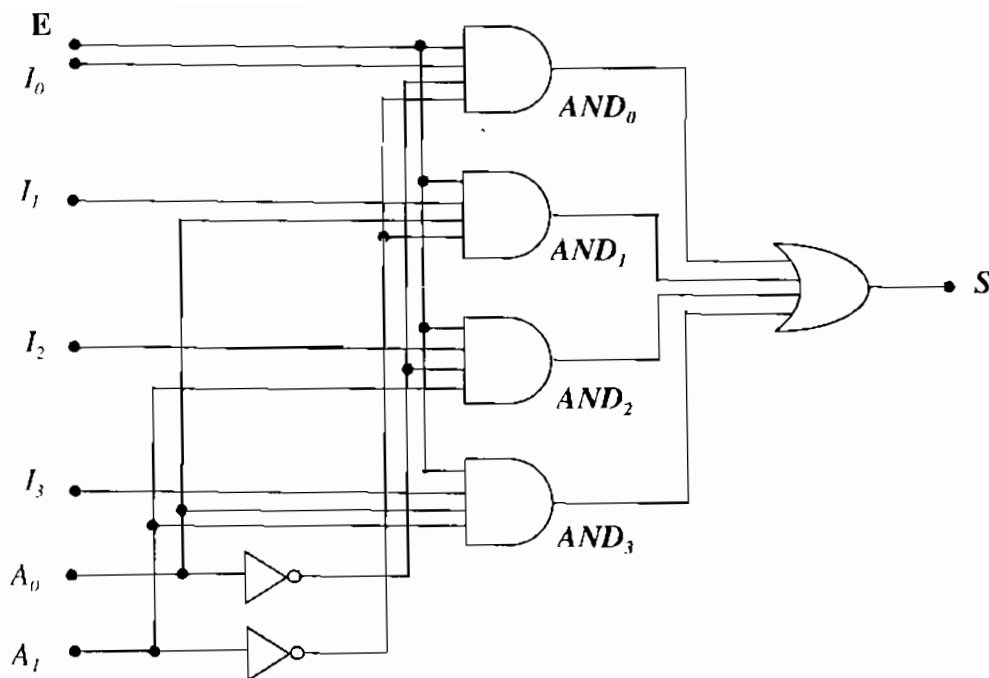
Bảng 4.1. Bảng chân lý bộ dồn kênh bốn bit

Các đầu vào				Đầu ra
E	A_1	A_0	I_i	S
0	X	X	X	0
1	0	0	I_0	(I_0)
1	0	0	I_1	(I_1)
1	1	0	I_2	(I_2)
1	1	1	I_3	(I_3)

Như bảng chân lý trên khi $E = 1$ và nếu $A_1, A_0 = 0,0$ thì đầu vào dữ liệu I_0 được chọn, dữ liệu ở I_0 (có thể = 0 hoặc = 1) được đưa tới đầu ra. Tương tự như vậy, đối với các trường hợp còn lại tương ứng với trạng thái của A_0 và A_1 trong bảng chân lý trên.

Hình 4.4 là sơ đồ logic bộ dồn kênh, được triển khai chi tiết từ sơ đồ khối 4.3. Ở sơ đồ này ta thấy có 4 cổng AND từ AND_0 đến AND_3 , mỗi cổng AND có 4 đầu vào. Đầu vào thứ nhất là E (Enable), đầu vào này dùng chung cho cả 4 cổng AND.

Đầu vào thứ hai là đầu vào dữ liệu, đây là đầu vào riêng rẽ cho từng cổng AND. Hai đầu vào còn lại của 4 cổng AND được lấy từ hai tín hiệu chọn A_0 và A_1 kết hợp với 2 cổng NOT để có thêm 2 tín hiệu A_0 đảo và A_1 đảo, tạo ra 4 tổ hợp tín hiệu chọn khác nhau hoàn toàn. Đầu ra của cả 4 cổng AND được đưa vào cổng OR và đầu ra bộ dồn kênh là đầu ra của cổng OR.



Hình 4.4: Sơ đồ logic bộ dồn kênh 4 bit

3. Nguyên lý làm việc của bộ dồn kênh 4 bit

Từ sơ đồ logic hình 4.4 ta có hàm đầu ra $S = AND_0 + AND_1 + AND_2 + AND_3$, và:

$$AND_0 = EI_0\bar{A}_1\bar{A}_0$$

$$AND_1 = EI_1\bar{A}_1A_0$$

$$AND_2 = EI_2A_1\bar{A}_0$$

$$AND_3 = EI_3A_1A_0$$

Như vậy $S = E(I_0\bar{A}_1\bar{A}_0 + I_1\bar{A}_1A_0 + I_2A_1\bar{A}_0 + I_3A_1A_0)$.

- Khi $E = 0$ thì $S = 0$, tức là bộ dồn kênh bị cấm, nó không làm việc.

- Khi $E = 1$ thì $S = I_0\bar{A}_1\bar{A}_0 + I_1\bar{A}_1A_0 + I_2A_1\bar{A}_0 + I_3A_1A_0$

Đến đây ta thấy đầu vào dữ liệu nào được chọn để lấy dữ liệu ra phụ thuộc vào tín hiệu chọn (các đầu vào A_1 và A_0 đảo).

Giả sử $A_1A_0 = 01$, như vậy chỉ có đầu ra của cổng $AND_1 = (I_1)$ còn các cổng AND khác đầu ra đều $= 0$, (vì cổng AND khi có một đầu vào $= 0$ là đầu ra $= 0$) tín hiệu từ I_1 đưa tới cổng OR và đầu ra của OR $= (I_1)$.

Các trường hợp khác tương tự.

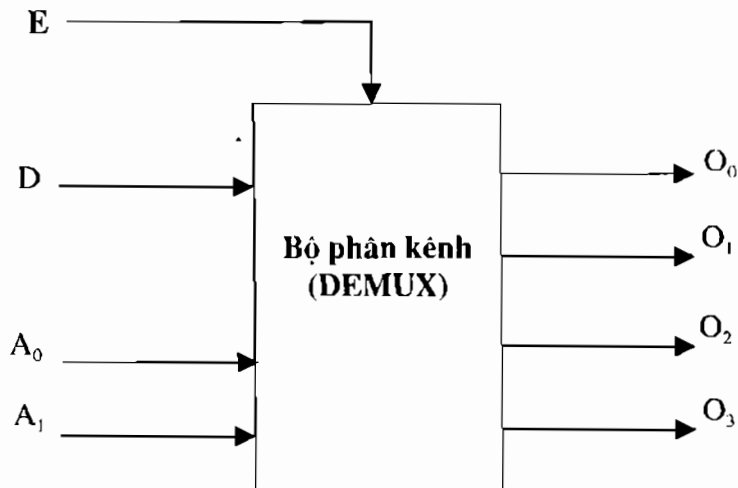
III. BỘ PHÂN KÊNH

1. Nhiệm vụ

Biến đổi thông tin vào nối tiếp thành thông tin ra song song.

Dưới sự điều khiển của n tín hiệu chọn, thực hiện sự chọn ra một đầu ra (một kênh) trong số 2^n đầu ra để đưa tín hiệu ở đầu vào (có một đầu vào), tới đầu ra được chọn.

Hình 4.5 là sơ đồ khối của bộ phân kênh bốn bit. Nó gồm một đầu vào cho phép E (Enable), một đầu vào dữ liệu D (Data), bốn đầu ra dữ liệu và hai đầu vào chọn A_0 . Với 2 đầu vào chọn cho phép chọn từng đầu ra riêng rẽ trong 4 đầu ra dữ liệu. (Quản lý được hết các đầu ra dữ liệu).



Hình 4.5: Sơ đồ khối bộ phân kênh 4 bit

2. Thiết kế bộ phân kênh 4 bit

Bộ dồn kênh cũng như bộ phân kênh đều là những mạch logic tổ hợp, vì vậy để thiết kế bộ phân kênh bốn bit ta thực hiện các bước của bài toán thiết kế mạch tổ hợp đã học ở chương 2.

Ở bộ phân kênh 4 bit, các bit dữ liệu từ một đầu vào được đưa lần lượt tới 4 đầu ra. Việc chọn từng đầu ra nhờ vào 2 đầu dẫn tín hiệu chọn đưa vào bộ phân kênh. Với hai đầu chọn sẽ có được 4 trạng thái tín hiệu khác nhau tương ứng cho 4 đầu ra của bộ phân kênh.

Với phân tích trên đây ta tiến hành lập bảng chân lý mô tả hoạt động của bộ phân kênh 4 bit.

2.1. Bảng chân lý

Một số quy ước:

- Giả sử $E = 1$, bộ phân kênh được phép làm việc.
- Dữ liệu ở đầu vào $D = 1111$ (4 bit liên tiếp bằng 1).
- Hai đầu vào chọn A_1, A_0 có 4 trạng thái khác nhau, mỗi trạng thái để chọn một đầu ra dữ liệu. Như vậy, ứng với mỗi trạng thái của A_1, A_0 chỉ chọn được một đầu ra dữ liệu duy nhất.
- Đầu ra dữ liệu nào được chọn = 1, đầu ra không được chọn = 0.

Bảng 4.2. Bảng chân lý của bộ phân kênh bốn bit

Các đầu vào				Các đầu ra			
E	A_1	A_0	D	O_0	O_1	O_2	O_3
1	0	0	1	1	0	0	0
1	0	1	1	0	1	0	0
1	1	0	1	0	0	1	0
1	1	1	1	0	0	0	1

2.2. Các hàm số

Dựa vào bảng chân lý 4.2 trên đây ta viết các hàm số đầu ra theo dạng chuẩn tắc tuyến.

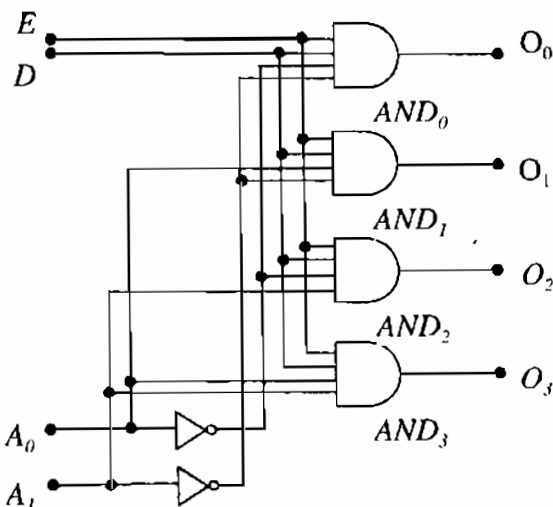
$$O_0 = E \bar{A}_1 \bar{A}_0 D$$

$$O_1 = E \bar{A}_1 A_0 D$$

$$O_2 = E A_1 \bar{A}_0 D$$

$$O_3 = E A_1 A_0 D$$

2.3. Sơ đồ logic



Hình 4.6: Sơ đồ logic bộ phân kênh 4 bit

Với 4 hàm số ở mục 2.2 trên ta thấy đây đều là các hàm thực hiện chức năng AND, mỗi hàm đều có 4 biến vào, trong đó có 2 biến chung cho cả 4 hàm đó là biến E và biến D. Có 2 biến sơ cấp là A_0 và A_1 từ hai biến này thực hiện chức năng NOT có thêm 2 biến A_0 đảo và A_1 đảo. Với những điều trên đây, để có được sơ đồ logic của bộ phân kênh 4 bit, ta dùng 4 cổng AND, mỗi cổng có 4 đầu vào để thực hiện chức năng hàm AND 4 biến, 2 cổng NOT để tạo ra các tín hiệu A_0 và A_1 đảo. E và D nối chung cho cả 4 cổng AND. Sơ đồ được thể hiện như hình 4.6.

IV. ỨNG DỤNG CỦA BỘ DỠN KÊNH VÀ PHÂN KÊNH

1. Mạch tạo hàm logic

Người ta có thể dùng mạch dõn kênh để thực hiện một hàm logic cho trong bảng chân lý. Quy tắc làm như sau: Nếu hàm logic có n biến độc lập thì phải chọn mạch dõn kênh có ít nhất n đầu vào chọn. Các biến độc lập trong bảng chân lý sẽ đặt ở các đầu vào chọn có trọng số tương ứng của MUX. Giá trị hàm logic trên các hàng của bảng chân lý sẽ ghi lên các đầu vào thông tin, có địa chỉ tương ứng với số thứ tự các hàng. Đầu ra S của mạch dõn kênh chính là hàm phụ thuộc trong bảng chân lý.

Hình 4.7.a là một ví dụ về bảng chân lý của hàm $F(C,B,A)$ và mạch dõn kênh được ghép nối để tạo hàm đã cho. Nếu mỗi đầu vào thông tin đều được nối với một khoá chuyển mạch K, như đã vẽ tượng trưng ở hình 4.7.b thì khi cần tạo hàm logic mới, ta chỉ việc đặt lại vị trí các khoá K, để thay đổi giá trị 0/1 logic trên các đầu vào thông tin cho phù hợp với bảng chân lý của hàm mới.

2. MUX được dùng làm bộ chọn dữ liệu hay chuyển mạch điện tử

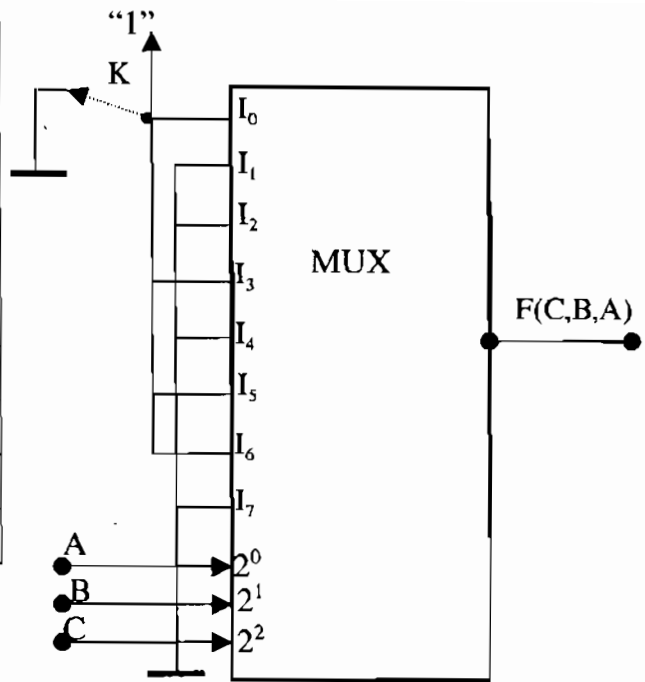
Dữ liệu vào được chọn để đưa tới đầu ra phụ thuộc vào tín hiệu điều khiển hay tín hiệu chọn. Điều này được miêu tả như trong hình 4.8.

- Số các đường dữ liệu vào có thể tăng lên nhờ sử dụng MUX có nhiều đầu vào hoặc bằng cách mắc tổ hợp nhiều MUX có số đầu vào nhỏ.

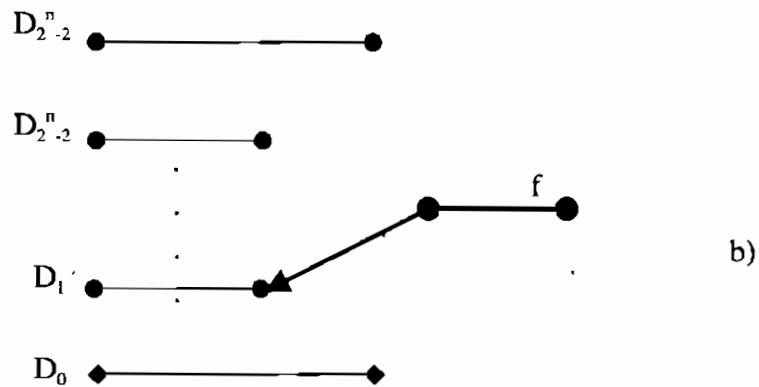
Ví dụ: Xây dựng MUX 8 đầu vào từ các MUX 4 đầu vào được minh hoạ ở hình 4.9. Việc điều khiển hoạt động của MUX này do một bộ đếm nhị phân 3 đầu ra thực hiện. Các đầu ra của bộ đếm được đưa vào N_1 , M_1 và M_2 như hình 4.9 thể hiện.

TT	Biến vào			Biến ra
	C	B	A	$F(C,B,A)$
0	0	0	0	1
1	0	0	1	0
2	0	1	0	0
3	0	1	1	1
4	1	0	0	0
5	1	0	1	1
6	1	1	0	1
7	1	1	1	0

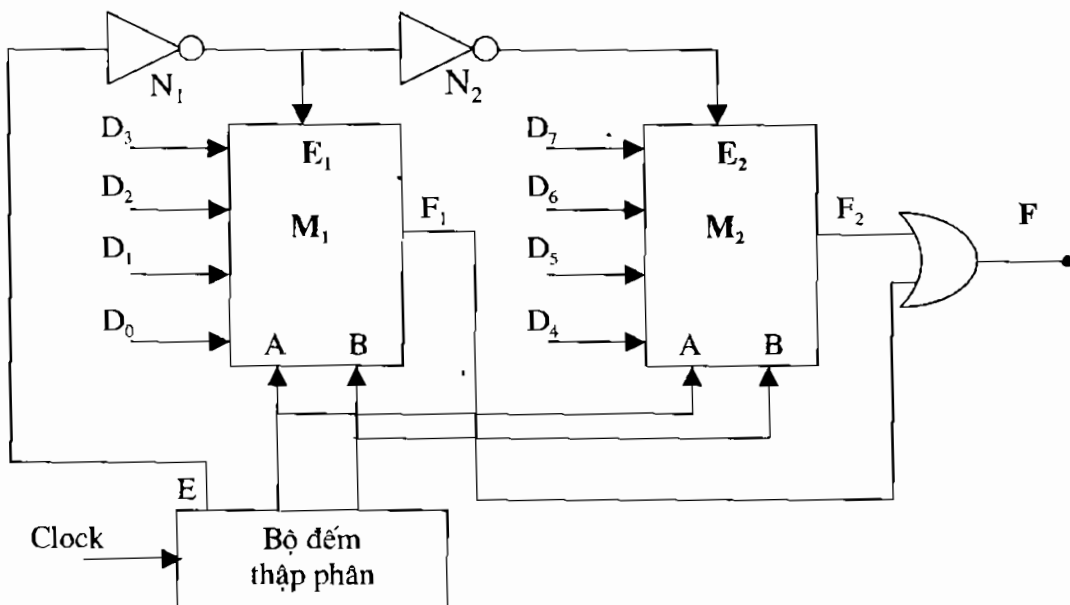
a)



Hình 4.7: Tạo hàm logic dùng MUX



Hình 4.8: Bộ chọn dữ liệu dùng MUX



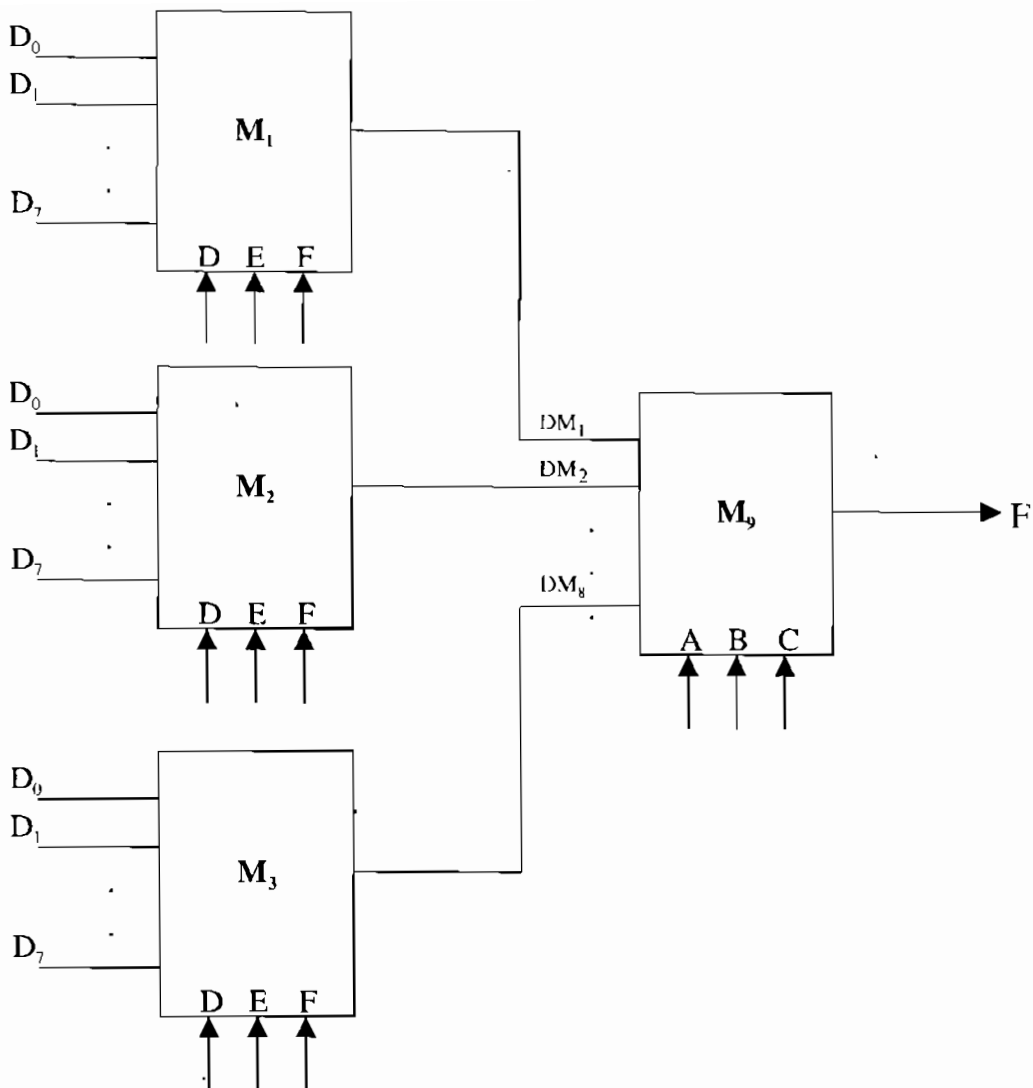
Hình 4.9: Tăng số đầu vào của MUX

Hình 4.9 thể hiện rõ khi $E = 0$ thì $E_1 = 1$ và $E_2 = 0$ và như vậy M_1 được phép làm việc, M_2 bị cấm. Lúc này tùy thuộc vào trạng thái của A và B mà một trong các đầu D_0 đến D_3 được chọn và dữ liệu ở đầu được chọn đó được đưa tới đầu ra F.

Tương tự, khi $E = 1$, M_2 được chọn, M_1 bị cấm. Tùy thuộc vào trạng thái của A và B mà một trong các đầu D_4 đến D_7 được chọn và dữ liệu ở đầu được chọn đó được đưa tới đầu ra F.

Ví dụ: Xây dựng MUX 64 đầu vào từ các MUX 8 đầu vào được mô tả trong hình 4.10.

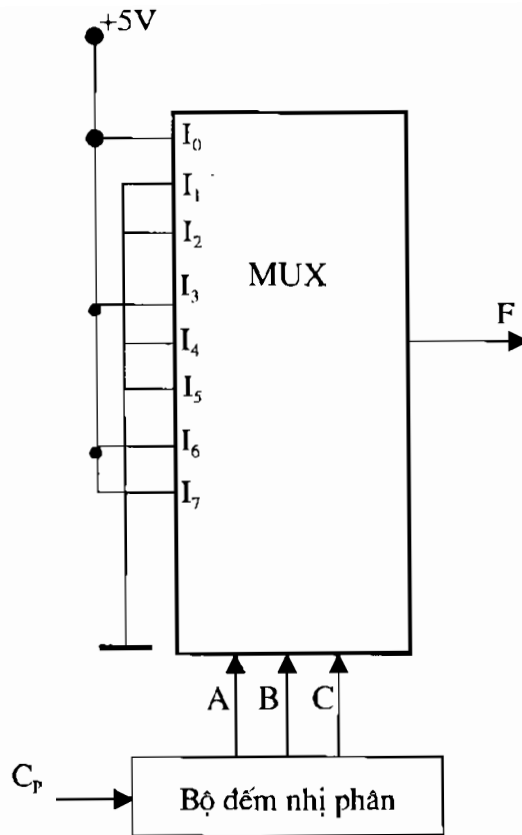
Các trạng thái tín hiệu ở 3 đầu vào D, E và F của cả 8 bộ dồn kênh M_1 đến M_8 hoàn toàn giống nhau. Như vậy, ứng với một trạng thái của D, E và F cả 8 đầu ra của cả 8 bộ dồn kênh đều có dữ liệu. Các dữ liệu này đều đưa vào M_9 . Tuy nhiên, việc chọn đưa dữ liệu ở đầu vào nào để cho ra đầu F còn phụ thuộc vào trạng thái của 3 đầu chọn A, B và C.



Hình 4.10: Tạo MUX64 đầu vào từ 9 bộ MUX8 đầu vào

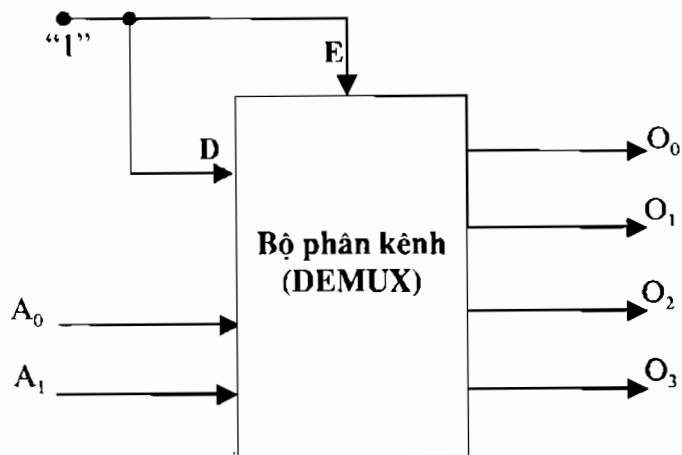
3. Tạo dãy tín hiệu nhị phân tuần hoàn

Muốn tạo dãy tín hiệu nhị phân tuần hoàn ta chỉ cần nối các đầu vào của MUX sử dụng với các mức logic nhất định. Việc thực hiện đưa tín hiệu ra nhờ một bộ đếm nhị phân mà các đầu ra của bộ đếm nhị phân này được đưa vào các đầu vào điều khiển của MUX đó. Ví dụ, tạo dãy tín hiệu nhị phân tuần hoàn 10010011 dùng MUX 8 đầu vào và bộ đếm nhị phân 3 bit như hình 4.11.



Hình 4.11: Dùng MUX 8 đầu vào để tạo dãy tín hiệu nhị phân tuần hoàn 10010011

4. Dùng DEMUX làm bộ giải mã



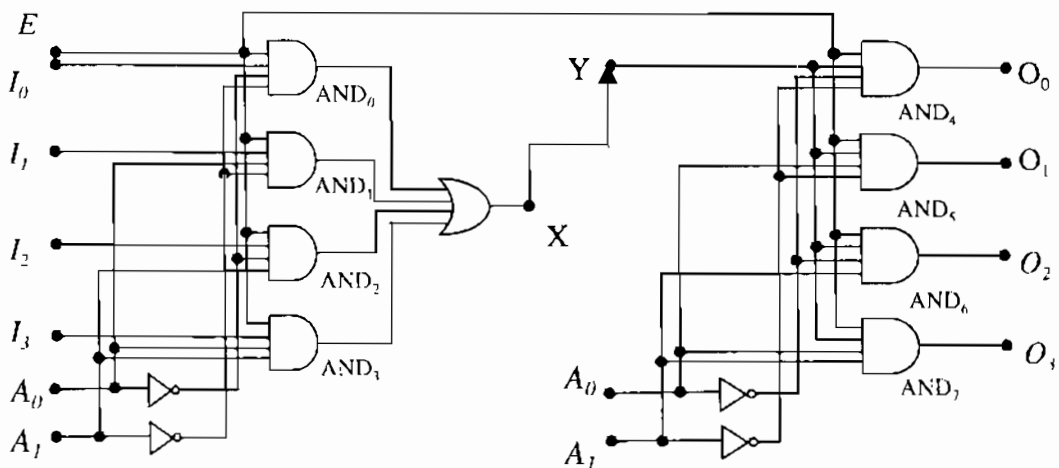
Hình 4.12: Sơ đồ khối bộ phân kênh dùng làm bộ giải mã 4 bit

Từ sơ đồ khối của DEMUX hình 4.12, khi E và D = 1, tùy theo trạng thái của các đầu vào A_i, từng đầu ra tương ứng sẽ được chọn và khi được chọn nó sẽ có giá trị bằng “1”, các đầu ra khác khi đó không được chọn có giá trị bằng “0”.

DEMUX khi làm việc như một bộ giải mã có đặc điểm: mỗi đầu ra là một tích đầy đủ n biến đầu vào. $O_i = E.D.A_i$

5. Truyền thông tin song song

Hình 4.13 là sơ đồ logic truyền thông tin song song 4 bit. Thông tin song song 4 bit được đặt vào các đầu I_i tương ứng. Tín hiệu điều khiển A_i sẽ chọn lần lượt từng đầu vào và đầu vào nào được chọn thông tin ở đầu vào đó được truyền tới đầu ra O_i tương ứng. Việc đưa được thông tin ra đúng đầu ra O_i tương ứng là do A_i.



Hình 4.13: Sơ đồ logic mạch truyền tin song song 4 bit

Câu hỏi và bài tập

1. Phân tích nguyên lý làm việc của bộ dồn kênh bốn bit.
2. Xây dựng sơ đồ logic và phân tích hoạt động bộ dồn kênh tám bit.
3. Dùng các bộ dồn kênh bốn bit để xây dựng bộ dồn kênh mười sáu bit.
4. Dùng bộ dồn kênh để thực hiện các hàm số sau:
 - a) $F = \overline{A} \overline{B} C + \overline{A} B \overline{C} + \overline{A} B C$
 - b) $F = \overline{A} B \overline{C} + A \overline{B} C + A B \overline{C} + A B C$
5. Phân tích nguyên lý làm việc của bộ phân kênh bốn bit.
6. Xây dựng sơ đồ logic và phân tích hoạt động bộ phân kênh tám bit.
7. Xây dựng mạch truyền thông tin song song tám bit từ các bộ dồn kênh và phân kênh.

Chương 5

CÁC BỘ MÃ HOÁ VÀ GIẢI MÃ

Mục tiêu

- Hiểu và phân tích được nguyên lý hoạt động của các mạch mã hoá và giải mã thông dụng.
- Vận dụng kiến thức về các bộ mã hoá và giải mã để giải thích hoạt động của các mạch trong các thiết bị chuyên ngành.
- Rèn luyện được khả năng tư duy logic và yêu thích ngành nghề chuyên môn theo học.

Nội dung

- Xây dựng sơ đồ và phân tích nguyên lý làm việc của các bộ mã hoá và giải mã.
- Phân tích hoạt động của bộ giải mã thông dụng (BCD ra mã bảy thanh).

I. KHÁI NIỆM CHUNG

Nói một cách khái quát, mã hoá là dùng văn tự, kí hiệu hay mã để biểu thị một đối tượng xác định. Có thể thấy biết bao ví dụ về mã hoá trong đời sống như đặt tên cho trẻ sơ sinh, mỗi vận động viên được mang một số trên trang phục trong thi đấu. Tên trẻ là văn tự, vận động viên được mang số theo hệ đếm thập phân. Văn tự và hệ đếm thập phân không tiện dùng cho mạch số. Mã hoá nhị phân là quá trình dùng mã nhị phân để biểu thị đối tượng xét đến (đối tượng này là tín hiệu). Mã nhị phân chỉ có hai chữ số 0 và 1, dễ dàng biểu thị bằng trạng thái mạch điện, nên được dùng rộng rãi trong mạch số. Biểu thị số lượng nhiều thì tăng số bit (Binary digit) trong cách viết số dựa theo vị trí. Mã nhị phân 1 bit có 2 trạng thái (0, 1) tương ứng 2 tín hiệu. Mã nhị phân 2 bit có 4 trạng thái (00, 01, 10, 11) tương ứng 4 tín hiệu. Tổng quát mà nói, mã nhị phân n bit có 2^n trạng thái, có thể biểu thị 2^n tín hiệu. Vậy, để tiến hành mã hoá N tín hiệu cần sử dụng n bit, theo công thức $2^n \geq N$.

Bộ mã hoá là mạch điện thực hiện thao tác mã hoá. Căn cứ vào yêu cầu và đặc điểm khác nhau của tín hiệu được mã hoá, chúng ta có các bộ mã hoá

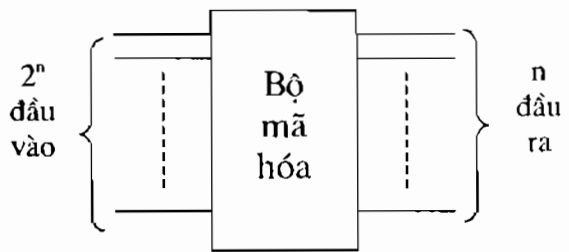
khác nhau, như bộ mã hoá nhị phân, bộ mã hoá nhị - thập phân, bộ mã hoá ưu tiên v.v.

Mỗi khi mã hoá, mỗi từ mã nhị phân đều được gán một hàm ý xác định, tức là mỗi từ mã biểu thị một tin tức hoặc một đối tượng xác định. Giải mã là quá trình phiên dịch hàm ý đã gán cho từ mã. Mạch điện thực hiện việc giải mã được gọi là bộ giải mã. Vậy, bộ giải mã phiên dịch từ mã thành tín hiệu đầu ra, biểu thị tin tức vốn có. Tùy theo yêu cầu sử dụng tin tức được giải mã, tín hiệu đầu ra có thể là xung hay mức điện áp. Có nhiều loại bộ giải mã, nhưng chúng đều có nguyên lý công tác và phương pháp thiết kế tương tự nhau.

II. BỘ MÃ HOÁ

1. Bộ mã hoá nhị phân

Bộ mã hoá nhị phân là mạch điện dùng n bit để mã hoá $N = 2^n$ tín hiệu. Chúng ta xét ví dụ dưới đây để thuyết minh nguyên lý công tác và quá trình thiết kế bộ mã hoá nhị phân.



Hình 5.1: Sơ đồ khối bộ mã hoá

2^n đầu vào, n đầu ra

Ví dụ: Hãy thiết kế bộ mã hoá thực hiện mã hoá 8 tín hiệu $Y_0, Y_1, \dots, Y_7$ theo mã nhị phân.

Bài giải:

Đối tượng được mã hoá là 8 tín hiệu đầu vào, tức là $Y_0, Y_1, \dots, Y_7$. Căn cứ vào công thức $N = 2^n = 8$ ta thấy đầu ra là mã nhị phân $n = 3$ bit, dùng A, B, C biểu thị.

1.1. Kê bảng chân lý

Việc mã hoá được tiến hành chỉ với 1 tín hiệu đầu vào ở một thời điểm. Ở đầu vào không cho phép có đồng thời từ 2 tín hiệu trở lên là logic 1, tức là $Y_0, Y_1, \dots, Y_7$ không cùng nhau. Vậy quan hệ logic giữa đầu ra với đầu vào có thể biểu thị bằng bảng chân lý hay bảng mã hoá.

Bảng 5.1. Bảng mã hóa tám tín hiệu

	C	B	A
Y_0	0	0	0
Y_1	0	0	1
Y_2	0	1	0
Y_3	0	1	1
Y_4	1	0	0
Y_5	1	0	1
Y_6	1	1	0
Y_7	1	1	1

a) Phương án 1

	C	B	A
Y_0	0	0	0
Y_1	0	0	1
Y_2	0	1	1
Y_3	0	1	0
Y_4	1	1	0
Y_5	1	1	1
Y_6	1	0	1
Y_7	1	0	0

b) Phương án 2

Dùng mã nhị phân 3 bit biểu thị 8 tín hiệu đầu vào có thể có nhiều phương án khác nhau, bảng 5.1 giới thiệu 2 phương án. Phương án 1 theo số đếm nhị phân. Phương án 2 là mã Gray 3 bit.

1.2. Hàm số logic

$Y_0 \div Y_7$ là không đồng thời. Để có biểu thức của hàm, chỉ cần lấy các biến làm cho hàm số bằng 1 cộng lại, ta có biểu thức hàm đã tối thiểu. Từ bảng 5.1.a (phương án 1) ta có biểu thức của hàm số:

$$C = Y_4 + Y_5 + Y_6 + Y_7$$

$$B = Y_2 + Y_3 + Y_6 + Y_7$$

$$A = Y_1 + Y_3 + Y_5 + Y_7$$

Từ bảng 5.1.b (phương án 2) ta có biểu thức hàm số:

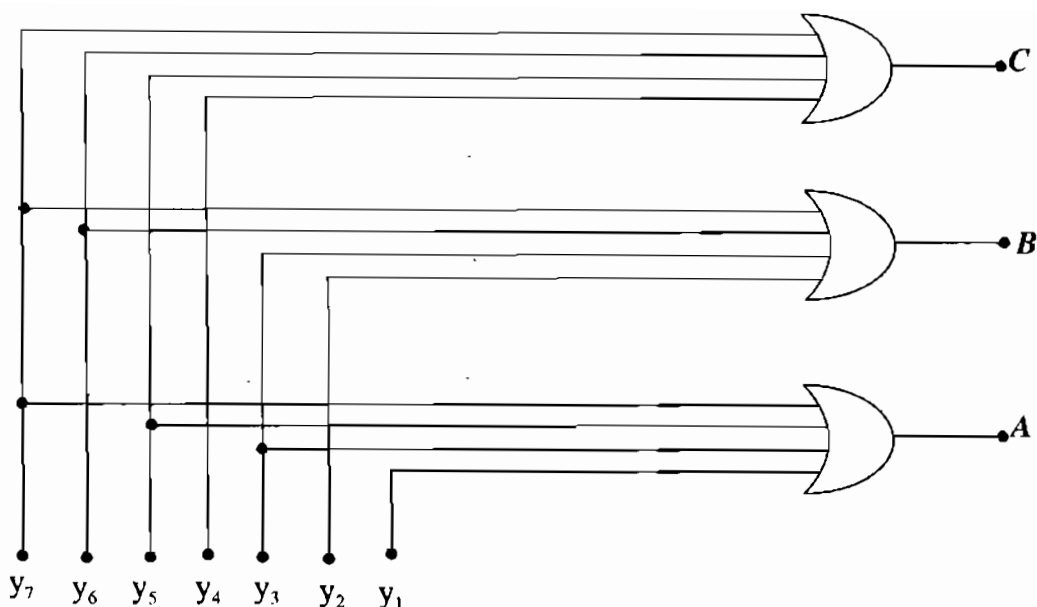
$$C = Y_4 + Y_5 + Y_6 + Y_7$$

$$B = Y_2 + Y_3 + Y_4 + Y_5$$

$$A = Y_1 + Y_2 + Y_5 + Y_6$$

1.3. Vẽ sơ đồ logic

Sơ đồ logic theo các hàm số ở phương án 1:



Hình 5.2: Bộ mã hoá nhị phân 3 bit

Cần lưu ý rằng, trong sơ đồ logic hình 5.2, sự mã hoá của Y_0 được ngầm định: khi $Y_1 \div Y_7$ đều bằng 0, trạng thái đầu ra của mạch (CBA = 000) là mã của Y_0 .

Trong bài toán phân tích, từ sơ đồ logic bộ mã hoá đã cho, ta phân tích mối quan hệ logic giữa đầu vào và đầu ra, tuần tự cho tín hiệu đầu vào đến bộ mã hoá, rồi căn cứ vào mức logic (0 hoặc 1) của các đầu ra mạch điện, tìm ra mã tương ứng, liệt kê thành bảng mã hoá.

2. Bộ mã hoá thập phân sang mã nhị phân

Bộ mã hoá thập phân sang mã nhị phân hay gọi tắt là bộ mã hoá thập phân, là mạch điện chuyển mã hệ thập phân bao gồm 10 chữ số 0, 1, 2, 3, 4, 5, 6, 7, 8, 9 thành mã nhị phân.

Đầu vào là 10 chữ số, đầu ra là nhóm mã số nhị phân, gọi là mã nhị - thập phân (BCD - Binary Coded Decimal). Căn cứ vào công thức $2^n \geq N = 10$, ta có $n = 4$. Mã nhị phân 4 bit có 16 trạng thái (từ mã). Chỉ cần chọn 10 từ mã tùy ý trong số đó là đủ để biểu thị 10 tín hiệu đầu vào. Vậy có rất nhiều phương án. Cùng với lý do như đã biết ở ví dụ 1, 10 tín hiệu đầu vào tồn tại độc lập nhau.

Nguyên lý công tác và quá trình thiết kế của bộ mã hoá nhị - thập phân tương tự như bộ mã hoá nhị phân xét ở ví dụ trên. Dưới đây lấy mã BCD 8421 làm ví dụ để thuyết minh. Bảng 5.2 là bảng mã hoá BCD 8421 rất thường dùng.

Bảng 5.2. Bảng mã hóa thập phân ra mã BCD

Đầu vào	Đầu ra			
Số thập phân	D	C	B	A
0(Y_0)	0	0	0	0
1(Y_1)	0	0	0	1
2(Y_2)	0	0	1	0
3(Y_3)	0	0	1	1
4(Y_4)	0	1	0	0
5(Y_5)	0	1	0	1
6(Y_6)	0	1	1	0
7(Y_7)	0	1	1	1
8(Y_8)	1	0	0	0
9(Y_9)	1	0	0	1

Từ bảng 5.2 ta viết được các hàm số:

$Y_0 \div Y_9$ là không đồng thời. Chỉ cần lấy các biến nào làm cho hàm số đầu ra bằng 1 cộng lại thì ta có biểu thức tối thiểu hoá. Từ bảng 5.2 ta có các biểu thức hàm số đầu ra dưới đây:

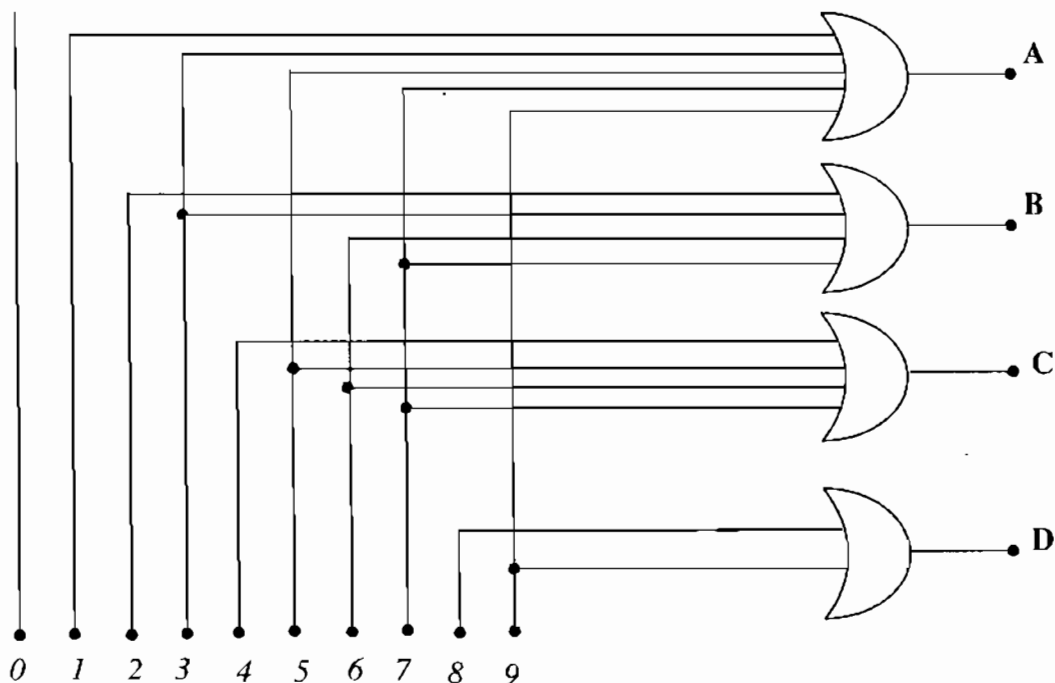
$$D = Y_8 + Y_9$$

$$C = Y_4 + Y_5 + Y_6 + Y_7$$

$$B = Y_2 + Y_3 + Y_6 + Y_7$$

$$A = Y_1 + Y_3 + Y_5 + Y_7 + Y_9$$

Dựa vào các hàm số trên vẽ sơ đồ logic như hình 5.3



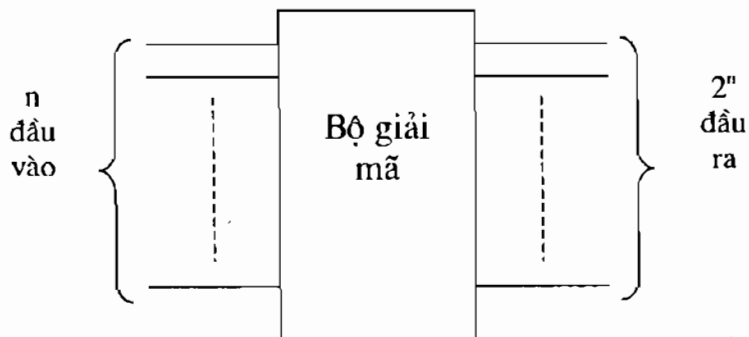
Hình 5.3: Bộ mã hóa thập phân ra mã BCD

III. BỘ GIẢI MÃ n ĐẦU VÀO 2^n ĐẦU RA

1. Nhiệm vụ

Bộ giải mã thực hiện chuyển đổi tín hiệu từ n đầu vào thành 2^n đầu ra.

Sơ đồ khối của bộ giải mã hình 5.4.



Hình 5.4: Sơ đồ khối bộ giải mã n đầu vào, 2^n đầu ra

2. Thiết kế bộ giải mã

Ta sẽ thiết kế bộ giải mã n đầu vào, 2^n đầu ra, với $n = 3$.

* *Phân tích yêu cầu:*

Đầu vào là nhóm từ mã nhị phân 3 bit. Đầu ra là tám tín hiệu tương ứng các từ mã.

* *Bảng chân lý:*

Ứng với mỗi từ mã đầu vào có một đầu ra được chọn và mức tích cực của nó khi được chọn = 1 logic. Ba đầu vào ký hiệu là C, B, A có tám đầu ra ký hiệu từ Y_0 đến Y_7 . (bảng 5.3)

Bảng 5.3. Bảng chân lý bộ giải mã ba đầu vào, tám đầu ra

Đầu vào			Đầu ra							
C	B	A	Y_0	Y_1	Y_2	Y_3	Y_4	Y_5	Y_6	Y_7
0	0	0	1	0	0	0	0	0	0	0
0	0	1	0	1	0	0	0	0	0	0
0	1	0	0	0	1	0	0	0	0	0
0	1	1	0	0	0	1	0	0	0	0
1	0	0	0	0	0	0	1	0	0	0
1	0	1	0	0	0	0	0	1	0	0
1	1	0	0	0	0	0	0	0	1	0
1	1	1	0	0	0	0	0	0	0	1

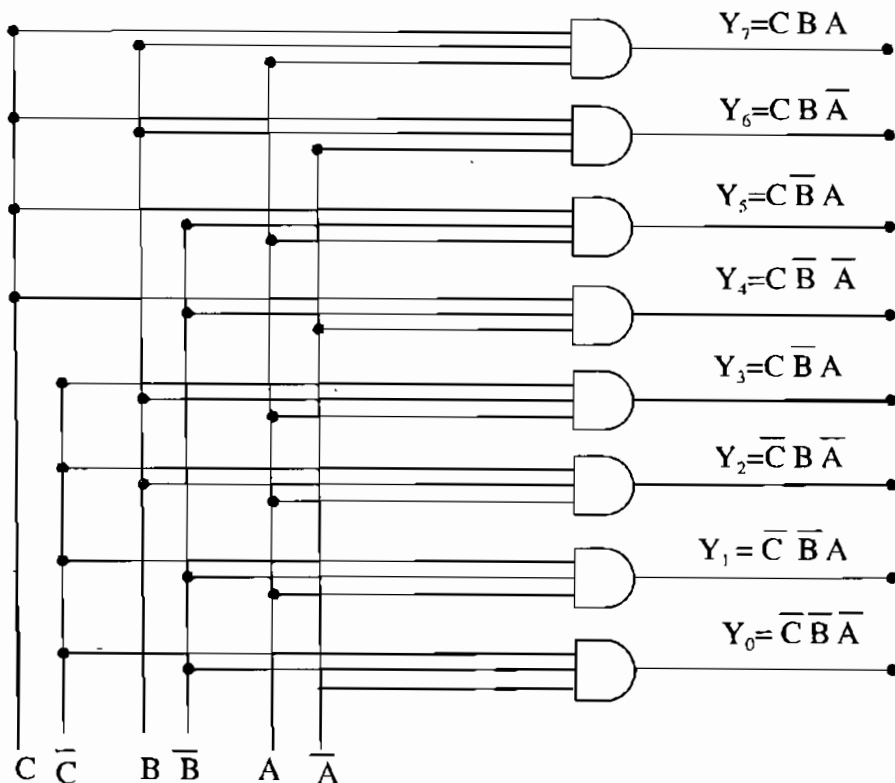
* *Các hàm số*

$$\begin{array}{llll}
 Y_0 = \overline{C} \overline{B} \overline{A} & Y_1 = \overline{C} \overline{B} A & Y_2 = \overline{C} B \overline{A} & Y_3 = \overline{C} B A \\
 Y_4 = C \overline{B} \overline{A} & Y_5 = C \overline{B} A & Y_6 = C B \overline{A} & Y_7 = C B A
 \end{array}$$

Các hàm trên đều thực hiện chức năng AND. Dựa vào các hàm số này, vẽ sơ đồ logic.

* *Sơ đồ logic*

Hình 5. 5 là sơ đồ logic của bộ giải mã 3 đầu vào 8 đầu ra.



Hình 5.5: Sơ đồ logic bộ giải mã 3 đầu vào, 8 đầu ra

IV. BỘ GIẢI MÃ BCD RA THẬP PHÂN

1. Nhiệm vụ

Thực hiện việc chuyển đổi mười từ mã BCD thành mười tín hiệu đầu ra tương ứng mười chữ số thập phân.

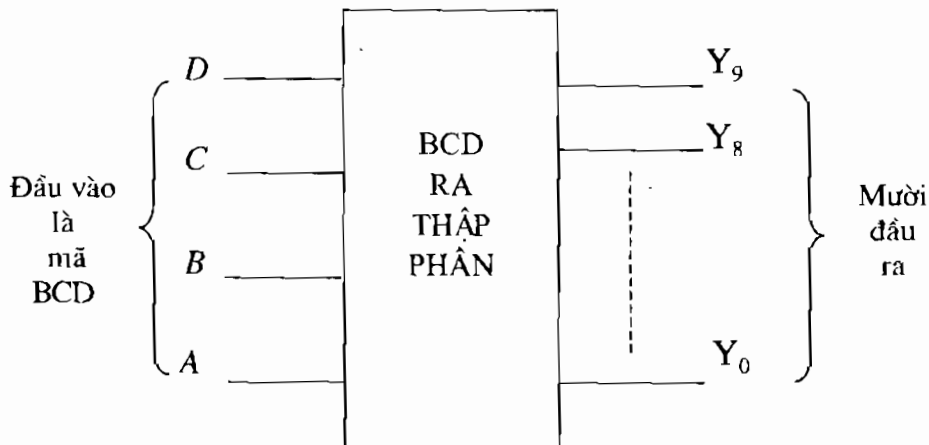
2. Thiết kế bộ giải mã BCD ra thập phân

Từ nhiệm vụ nêu trên chúng ta tìm hiểu bộ giải mã thông qua các bước thực hiện thiết kế mạch tổ hợp.

2.1. Phân tích nhiệm vụ

Đầu vào là mã BCD, mỗi từ mã của nó có 4 bit, như vậy cần 4 đầu vào bộ giải mã và kí hiệu các đầu vào là D, C, B, A tương ứng với 4 bit của mã BCD. Đầu ra có 10 tín hiệu tương ứng với 10 chữ số thập phân, vậy cần 10 đầu ra. Các đầu ra này được kí hiệu từ Y_0 đến Y_9 . Sơ đồ khối của bộ giải mã như hình 5. 6.

Ứng với mỗi từ mã ở đầu vào chỉ có một đầu ra được chọn và khi đó đầu ra được chọn được biểu diễn bằng 1 logic. Các đầu ra không được chọn ở mức 0 logic.



Hình 5.6: Sơ đồ khối bộ giải mã BCD ra thập phân

2.2. Bảng chân lý

Bảng 5.4. Bảng chân lý của bộ giải mã BCD ra thập phân

TT	Đầu vào				Đầu ra									
	D	C	B	A	Y ₀	Y ₁	Y ₂	Y ₃	Y ₄	Y ₅	Y ₆	Y ₇	Y ₈	Y ₉
1	0	0	0	0	1	0	0	0	0	0	0	0	0	0
2	0	0	0	1	0	1	0	0	0	0	0	0	0	0
3	0	0	1	0	0	0	1	0	0	0	0	0	0	0
4	0	0	1	1	0	0	0	1	0	0	0	0	0	0
5	0	1	0	0	0	0	0	0	1	0	0	0	0	0
6	0	1	0	1	0	0	0	0	0	1	0	0	0	0
7	0	1	1	0	0	0	0	0	0	0	1	0	0	0
8	0	1	1	1	0	0	0	0	0	0	0	1	0	0
9	1	0	0	0	0	0	0	0	0	0	0	0	1	0
10	1	0	0	1	0	0	0	0	0	0	0	0	0	1

2.3. Hàm số logic

Từ bảng chân lý trên, theo phương pháp chuẩn tắc tuyển viết được các hàm logic.

$$Y_0 = \overline{D} \overline{C} \overline{B} \overline{A}$$

$$Y_2 = \overline{D} \overline{C} B \overline{A}$$

$$Y_4 = \overline{D} C \overline{B} \overline{A}$$

$$Y_6 = \overline{D} C B \overline{A}$$

$$Y_8 = D \overline{C} \overline{B} \overline{A}$$

$$Y_1 = \overline{D} \overline{C} B A$$

$$Y_3 = \overline{D} \overline{C} \overline{B} A$$

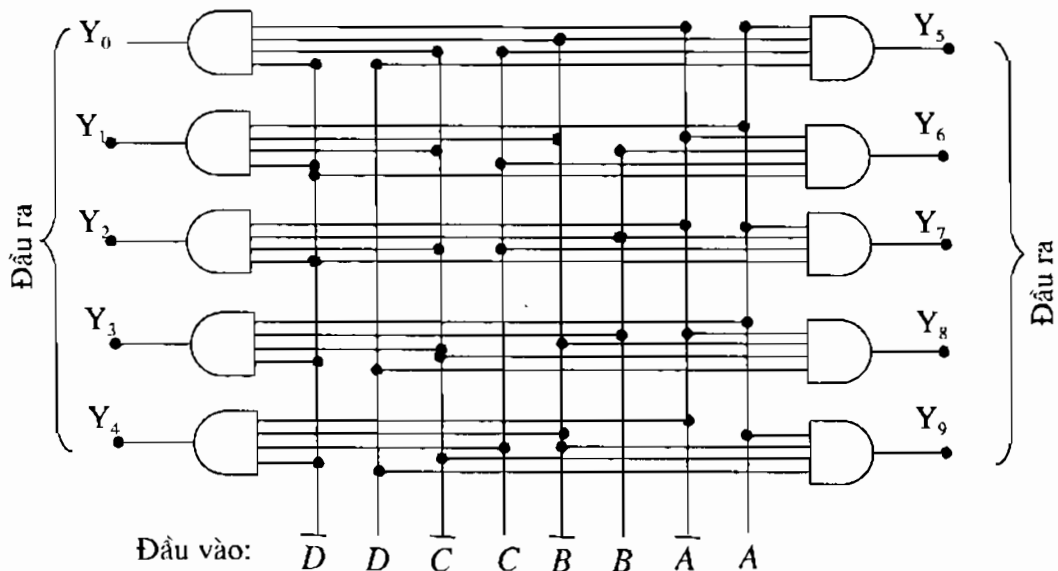
$$Y_5 = \overline{D} C \overline{B} A$$

$$Y_7 = \overline{D} C B A$$

$$Y_9 = D C B \overline{A}$$

Tất cả có 10 hàm logic, các hàm này đều thực hiện chức năng của hàm AND, mỗi hàm có 4 biến. Vậy, để có được sơ đồ logic ta dùng 10 cổng AND, mỗi cổng có 4 đầu vào.

2.4. Sơ đồ logic



Hình 5.7: Mạch giải mã BCD ra thập phân

3. Phân tích hoạt động

Sau khi đã thiết kế xong bộ giải mã BCD ra thập phân, để hiểu sâu hơn về bộ giải mã này ta tiến hành phân tích hoạt động của nó.

Theo bảng chân lý, khi đặt vào đầu vào bộ giải mã một từ mã là 0101 (tương ứng với số 5 ở hệ 10) ta thấy đầu ra $Y_5 = 1$ còn các đầu ra khác $= 0$ vì $Y_5 = \overline{D} C \overline{B} A$

$D = 0$ nên $\overline{D} = 1$

$$C = 1$$

$$B = 0 \text{ vậy } \overline{B} = 1$$

$$A = 1.$$

Một cổng AND có tất cả các đầu vào đều = 1 nên đầu ra = 1. Với phân tích như trên các cổng AND khác đều có ít nhất một đầu vào = 0 logic nên đầu ra của chúng = 0 logic. Bạn đọc có thể phân tích chi tiết điều này.

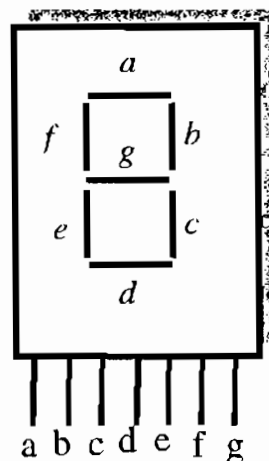
V. BỘ GIẢI MÃ BCD RA 7 THANH

1. Đèn hiển thị 7 thanh (7 segment)

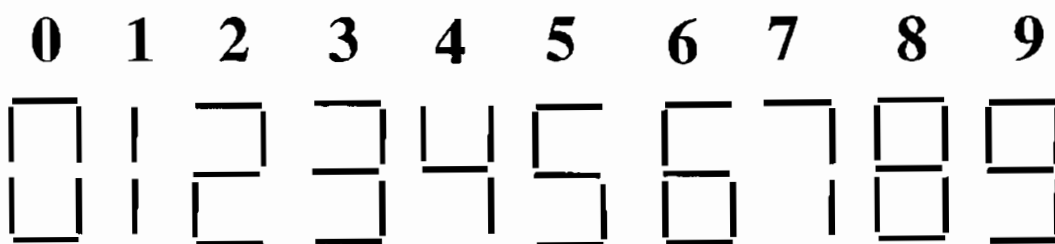
Trong các hệ thống số, để thể hiện 10 chữ số thập phân từ 0 đến 9 người ta dùng đèn hiển thị 7 thanh có hình dạng và tên gọi cụ thể từng thanh như hình 5.8.

Các chữ số thập phân được biểu diễn bằng đèn bảy thanh ở hình 5.9

Số 6 và số 9 có thể có hoặc không có thanh a, thanh d tùy theo loại vi mạch giải mã của từng hãng sản xuất.



Hình 5.8: Ký hiệu đèn hiển thị 7 thanh

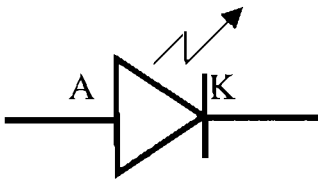


Hình 5.9: Biểu diễn số thập phân bằng đèn 7 thanh

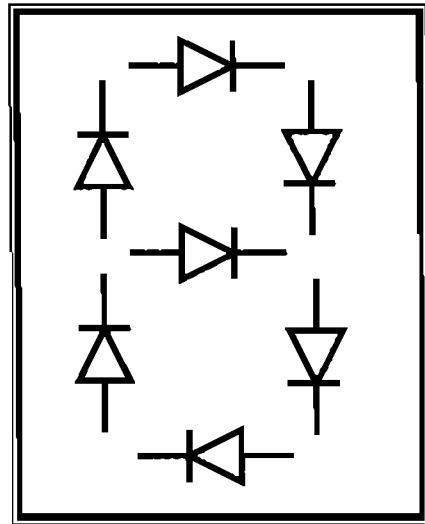
Hiện nay phổ biến hai phương thức hiển thị số bảy thanh:

- *Hiển thị bằng tinh thể lỏng*: (Liquid Crystal Display: LCD). Tinh thể lỏng có thể là một trong những hợp chất hữu cơ có tính chất quang học, và được đặt thành lớp giữa hai tấm kính với các điện cực trong suốt kết nối ở mặt trong. Bảy thanh tinh thể lỏng cũng ghép thành hình số tám và đặt trên một mặt nền BP (back plane). Bảy điện cực của 7 thanh được dẫn ra mặt nền, tạo thành đầu

ra chung gọi là cực nền BP. Nếu đặt lên cực nền và điện cực còn lại của thanh LCD hai dây xung vuông góc đối xứng, tần số 50 --> 60 Hz và lệch pha nhau 180° thì chất tinh thể lỏng được kích hoạt. Nó tán xạ ánh sáng ra mọi phía và thanh hiển thị tinh thể lỏng (LCD) sẽ nổi lên trên mặt nền. Khi không được kích hoạt, chất tinh thể lỏng cho ánh sáng đi qua và thanh LCD bị hoà lẫn vào mặt nền. Ưu điểm của hiển thị tinh thể lỏng là tiêu thụ dòng rất nhỏ, cả 7 thanh LCD loại nhỏ chỉ yêu cầu dòng khoảng $100\mu A$. Cần chú ý là nếu đặt điện áp một chiều chỉ cỡ trên 500mV trong vài phút lên hai điện cực thanh LCD cũng đủ để làm hỏng điện cực.



Hình 5.10: Ký hiệu LED



Hình 5.11: Đèn hiển thị 7 thanh dùng LED

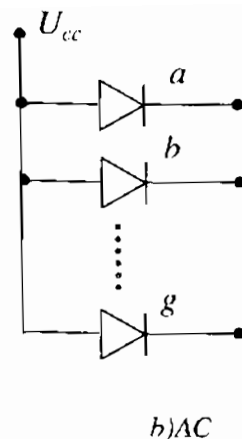
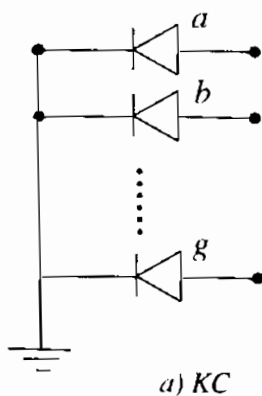
- *Hiển thị bằng diôt phát quang* (Light Emitting Diode: LED) (hình 5.10). Mỗi thanh trong đèn hiển thị 7 thanh là một LED. (Hình 5.11).

Khi $U_A > U_K$ có dòng điện chạy qua LED và nó phát sáng.

Có 2 loại đèn hiển thị 7 thanh dùng LED đó là loại Katot chung (Katot Common : KC) và Anot chung (AC: Anot Common).

- Đối với loại KC, (hình 5.12.a), đầu katot của cả bảy LED được nối chung lại. Tín hiệu điều khiển cho từng LED được đưa vào anot của từng LED.

- Ở loại AC, (hình 5.12.b), đầu anot của tất cả các LED được nối chung lại và nối với mức có điện áp cao, tương ứng với 1 logic (logic dương). Tín hiệu điều khiển cho từng LED được đưa vào katot của từng LED.



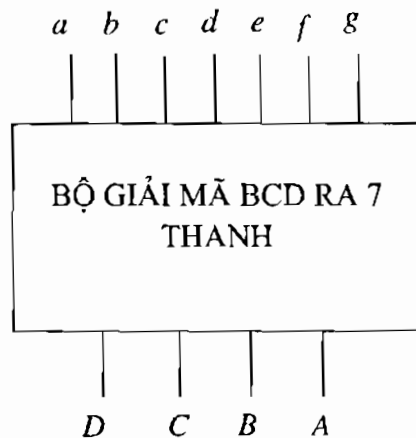
Hình 5.12: Đèn hiển thị LED 7 thanh mắc KC và AC.

2. Bộ giải mã BCD ra 7 thanh

Chúng ta sẽ thuyết minh nguyên lý công tác và quá trình thiết kế bộ giải mã hiển thị qua ví dụ bộ giải mã kích cho bộ hiển thị 7 thanh dùng LED. Do tồn tại hai loại hiển thị 7 thanh dùng LED là AC và KC nên có hai loại giải mã BCD ra mã 7 thanh tương ứng. Bộ giải mã được xây dựng theo bảng chân lý 5.5 là loại giải mã dùng cho đèn hiển thị KC. Đối với loại dùng cho hiển thị AC mức logic đảo lại. Bộ giải mã gồm 4 đầu vào để dẫn các từ mã BCD và có 7 đầu ra. Mỗi đầu ra được nối vào một đầu vào tương ứng của LED 7 thanh thông qua một điện trở hạn chế dòng. Như vậy có 7 đầu ra, sơ đồ khối bộ giải mã là hình 5.13.

2.1. Bảng chân lý

Đầu vào là mã BCD, ứng với mỗi từ mã đưa vào, các đầu ra sẽ tạo ra tín hiệu để điều khiển đèn hiển thị, hiển thị đúng số thập phân tương ứng với từ mã ở đầu vào.



Hình 5.13: Sơ đồ khối bộ giải mã BCD ra 7 thanh

Bảng 5.5. Bảng chân lý bộ giải mã BCD ra bảy thanh

TT	Đầu vào				Đầu ra						
	D	C	B	A	a	b	c	d	e	f	g
1	0	0	0	0	1	1	1	1	1	1	0
2	0	0	0	1	0	1	1	0	0	0	0
3	0	0	1	0	1	1	0	1	1	0	1
4	0	0	1	1	1	1	1	1	0	0	1
5	0	1	0	0	0	1	1	0	0	1	1
6	0	1	0	1	1	0	1	1	0	1	1
7	0	1	1	0	1	0	1	1	1	1	1
8	0	1	1	1	1	1	1	0	0	0	0
9	1	0	0	0	1	1	1	1	1	1	1
10	1	0	0	1	1	1	1	1	0	1	1

2.2. Hàm số

Từ bảng chân lý trên đây, lập bảng Karnaugh cho 7 hàm số.

BA \ DC					
		00	01	11	10
DC	00	1	0	1	1
	01	0	1	1	1
	11	X	X	X	X
	10	1	1	X	X

Hàm số a.

BA \ DC					
		00	01	11	10
DC	00	1	1	1	1
	01	1	0	1	0
	11	X	X	X	X
	10	0	0	1	1

Hàm số b.

BA \ DC					
		00	01	11	10
DC	00	1	1	1	0
	01	1	1	1	1
	11	X	X	X	X
	10	1	1	X	X

Hàm số c.

BA \ DC					
		00	01	11	10
DC	00	1	0	1	1
	01	0	1	0	1
	11	X	X	X	X
	10	1	1	X	X

Hàm số d.

		BA			
		00	01	11	10
DC	00	1	0	0	1
	01	0	0	0	1
	11	X	X	X	X
	10	1	0	X	X

Hàm số e.

		BA			
		00	01	11	10
DC	00	1	0	0	0
	01	1	1	0	1
	11	X	X	X	X
	10	1	1	X	X

Hàm số f.

		BA			
		00	01	11	10
DC	00	0	0	1	1
	01	1	1	0	1
	11	X	X	X	X
	10	1	1	X	X

Hàm số g.

Sau khi tối thiểu hoá các hàm này trên bảng Karnaugh ta được kết quả:

$$a = \overline{D} + \overline{B} + \overline{C}A + \overline{C}\overline{A}$$

$$b = \overline{C} + \overline{B}A + \overline{B}\overline{A}$$

$$c = \overline{C} + \overline{B} + \overline{A}$$

$$d = \overline{D} + \overline{C}\overline{B} + \overline{B}\overline{A} + \overline{C}\overline{A} + \overline{C}B\overline{A}$$

$$e = \overline{C}A + \overline{B}A$$

$$f = \overline{D} + \overline{C}\overline{B} + \overline{C}A + \overline{B}\overline{A}$$

$$g = \overline{D} + \overline{C}\overline{B} + \overline{C}B + \overline{B}\overline{A}$$

Từ các hàm đã tối thiểu trên đây, lập sơ đồ logic bằng cách dùng các phần tử tổ hợp.

Câu hỏi và bài tập

1. Trình bày khái niệm về mã hoá và giải mã.
2. Phân tích hoạt động của bộ mã hoá 2^n đầu vào n đầu ra.
3. Phân tích hoạt động của bộ mã hoá thập phân ra mã BCD.
4. Xây dựng và phân tích hoạt động của bộ giải mã BCD ra thập phân.
5. Trình bày về nguyên tắc dùng đèn hiển thị bảy thanh để hiển thị các chữ số thập phân.
6. Trình bày về cấu tạo của hai loại đèn hiển thị bảy thanh là AC và KC.
7. Xây dựng sơ đồ logic bộ giải mã ba đầu vào, tám đầu ra.
8. Lập bảng chân lý bộ giải mã BCD ra 7 thanh dùng cho loại đèn hiển thị AC và biểu diễn trên bảng Karnaugh.

Chương 6

CÁC PHẦN TỬ NHỚ CƠ BẢN

Mục tiêu

- Phân tích được nguyên tắc làm việc của các loại Flip-Flop (FF).
- Dựa vào nguyên tắc hoạt động của các loại Flip - Flop, vẽ được biểu đồ thời gian làm việc của nó.
- Rèn luyện được khả năng tư duy và sáng tạo trong công việc.

Nội dung trọng tâm

- Khái niệm về mạch dây và FF.
- Định nghĩa, phân loại và phân tích hoạt động của các loại FF.

Như đã trình bày ở chương 2, mạch số được chia làm 2 loại:

1. *Mạch tổ hợp (Combinational)* là mạch mà trị số ổn định của tín hiệu đầu ra ở thời điểm bất kỳ chỉ phụ thuộc vào tổ hợp các giá trị tín hiệu đầu vào ở thời điểm đó. Các phần tử cơ bản để xây dựng nên hệ tổ hợp là các mạch AND, OR, NAND, NOR,...

2. *Mạch dây - Mạch tuần tự (Sequential circuits)* là mạch có tín hiệu ra phụ thuộc không những vào tín hiệu vào mà còn phụ thuộc vào trạng thái trong của mạch, nghĩa là mạch có lưu trữ, nhớ các trạng thái. Để xây dựng mạch dây ngoài các mạch tổ hợp cơ bản như AND, OR, NAND, NOR... còn cần phải có các phần tử nhớ.

Chương này sẽ nghiên cứu các phần tử nhớ cơ bản để tạo thành mạch dây. Các mạch này được gọi là Flip - Flop (thường được viết tắt là FF), chúng là các phần tử nhớ đơn bit vì có khả năng nhớ được một chữ số nhị phân.

I. KHÁI NIỆM VỀ MẠCH DÂY VÀ FF

1. Khái niệm

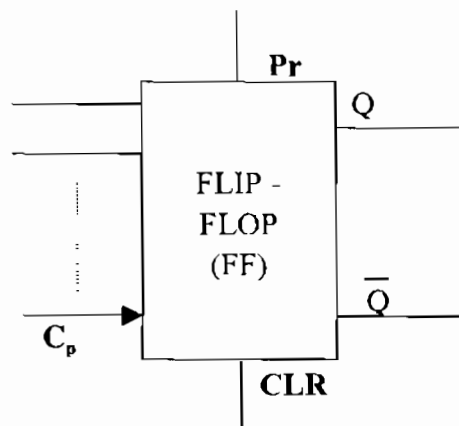
Flip - Flop (FF) là phần tử có khả năng lưu trữ (nhớ) một trong hai trạng thái 0 hay 1.

FF có từ 1 đến một vài đầu điều khiển, có 2 đầu ra luôn luôn ngược nhau là Q và $\bar{Q}$. Tùy từng loại FF, do chế tạo có thể còn có đầu vào xoá (thiết lập "0" - Clear), đầu vào thiết lập "1" (Preset).

Ngoài ra, FF còn thường hay có đầu vào đồng bộ (Clock).

Sơ đồ khối tổng quát của một FF được cho ở hình 6-1.

Các đầu vào điều khiển



Hình 6.1: Sơ đồ tổng quát cho một Flip - Flop

Các ký hiệu về tính tích cực được chỉ ra trong bảng 6-1.

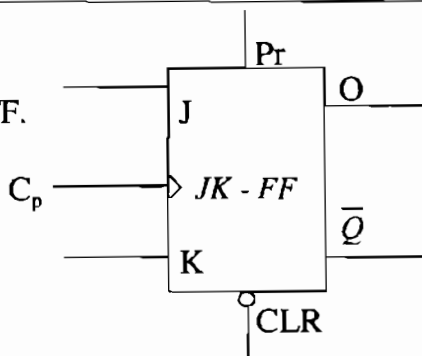
Ví dụ: Flip - Flop cho trong hình 6-2 là Flip - Flop loại JK, có đầu vào Pr (thiết lập "1"), tích cực ở "H", Clear (thiết lập "0"), tích cực ở "L", C_p tích cực ở sườn dương.

Bảng 6.1: Bảng ký hiệu về tính tích cực

Ký hiệu	Tính tích cực của tín hiệu
	Tích cực là mức thấp (L)
	Tích cực là mức cao (H)
	Tích cực là sườn dương của xung nhịp
	Tích cực là sườn âm của xung nhịp

2. Phân loại

Có nhiều cách phân loại FF.



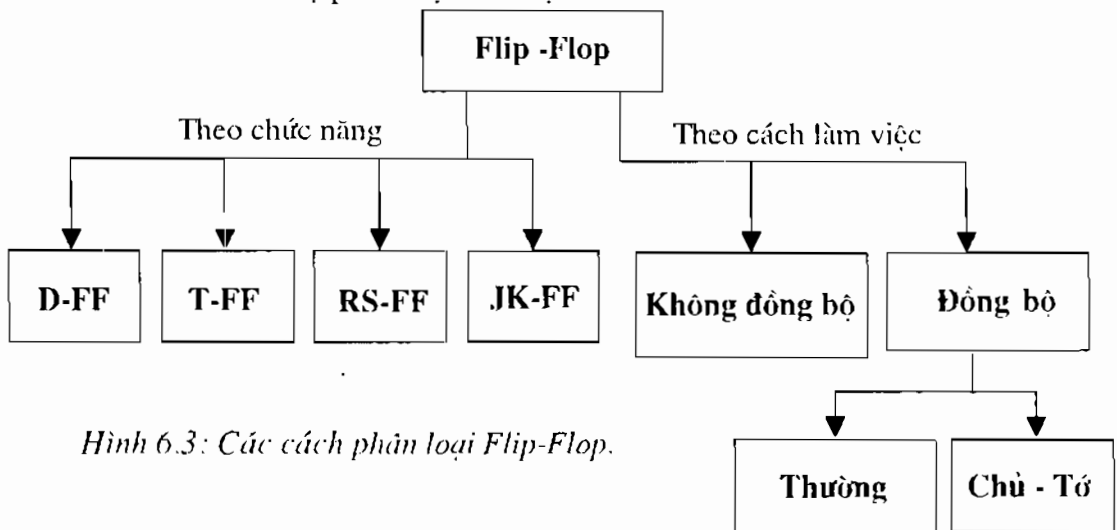
Hình 6.2: Mạch Flip - Flop (FF) (Ví dụ loại tín hiệu kích)

- Theo chức năng làm việc của các đầu vào điều khiển: Hiện nay thường sử dụng loại FF một đầu vào D-FF, T-FF và loại FF hai đầu vào RS-FF và JK-FF, ngoài ra đôi khi còn có thể gặp loại FF nhiều đầu vào.

- Theo cách làm việc ta có các loại FF:

Không đồng bộ và đồng bộ. Đối với loại không đồng bộ, các tín hiệu điều khiển vẫn điều khiển được hoạt động của FF đúng không cần tín hiệu đồng bộ. Ngược lại, ở loại FF đồng bộ các tín hiệu điều khiển chỉ điều khiển được hoạt động của FF khi và chỉ khi có tín hiệu đồng bộ và tín hiệu này tích cực. Loại đồng bộ này lại được chia làm loại đồng bộ thường và loại đồng bộ chủ - tớ (Master - Slave).

Sơ đồ khối của sự phân loại FF được cho ở hình 6-3.



Hình 6.3: Các cách phân loại Flip-Flop.

3. Các loại Flip - Flop (FF) và điều kiện đồng bộ

Theo chức năng có 4 loại FF cơ bản: D, T, RS và JK. Bảng 6-2 là bảng chân lý của các loại FF, với bảng 6.2.a cho D-FF, bảng 6.2.b cho T-FF, bảng 6.2c cho RS-FF và hình 6.2d cho JK-FF.

Bảng 6.2a: Bảng chân lý của D-FF

D	Q	Q'
0	0	0
0	1	0
1	0	1
1	1	1

Bảng 6.2b: Bảng chân lý của T-FF

T	Q	Q'
0	0	0
0	1	1
1	0	1
1	1	0

Bảng 6.2c: Bảng chân lý của RS-FF Bảng 6.2d: Bảng chân lý của JK-FF

R	S	Q	Q'
0	0	0	0
0	0	1	1
0	1	0	1
0	1	1	1
1	0	0	0
1	0	1	0
1	1	0	cấm
1	1	1	cấm

c)

J	K	Q	Q'
0	0	0	0
0	0	1	1
0	1	0	0
0	1	1	0
1	0	0	1
1	0	1	1
1	1	0	1
1	1	1	0

d)

* Cần chú ý đối với các loại FF một số vấn đề như sau:

- Các D-FF và RS-FF có thể làm việc ở chế độ không đồng bộ, vì với mỗi tập tín hiệu vào điều khiển D, RS luôn luôn tồn tại ít nhất 1 trong các trạng thái ổn định.

- Các T-FF và JK-FF không thể làm việc ở chế độ không đồng bộ, vì mạch sẽ rơi vào trạng thái dao động nếu như tập tín hiệu vào là 1 1 với JK-FF hoặc là 1 với T-FF.

Như vậy, các D-FF, RS-FF có thể làm việc ở cả 2 chế độ: đồng bộ và không đồng bộ còn các T-FF và JK-FF chỉ có thể làm việc ở chế độ đồng bộ.

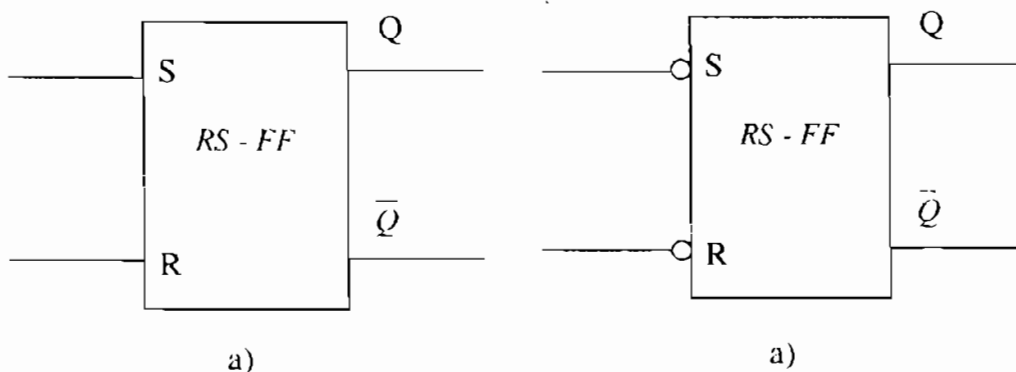
II. RS-FF

1. Định nghĩa

RS-FF là một FF có 2 đầu vào điều khiển R và S. S là đầu vào đặt (thiết lập $Q = "1"$ - Set) còn R là đầu vào xóa $Q = "0"$ (Reset).

2. Ký hiệu các loại RS- FF

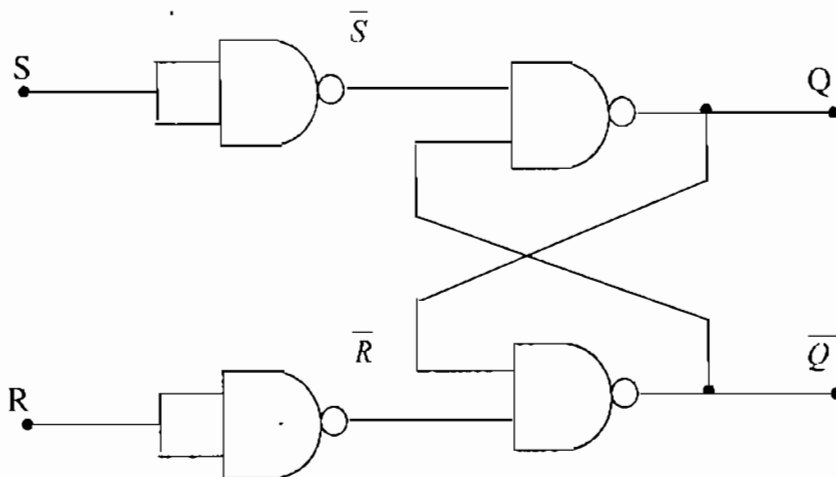
2.1. Loại hoạt động không đồng bộ



Hình 6.4: Flip - Flop RS hoạt động không đồng bộ

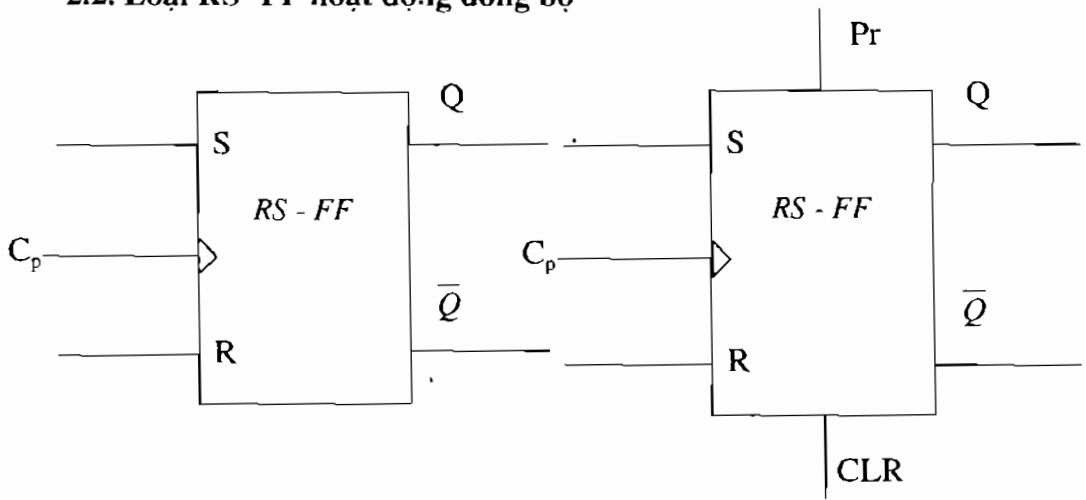
Loại RS - FF hoạt động không đồng bộ chỉ có hai đầu vào R và S. Hình 6.4.a, mức tích cực của tín hiệu điều khiển FF là mức H. Hình 6.4b, mức tích cực của tín hiệu điều khiển là L.

Dùng các cổng logic để xây dựng RS hoạt động không đồng bộ như hình 6.5.



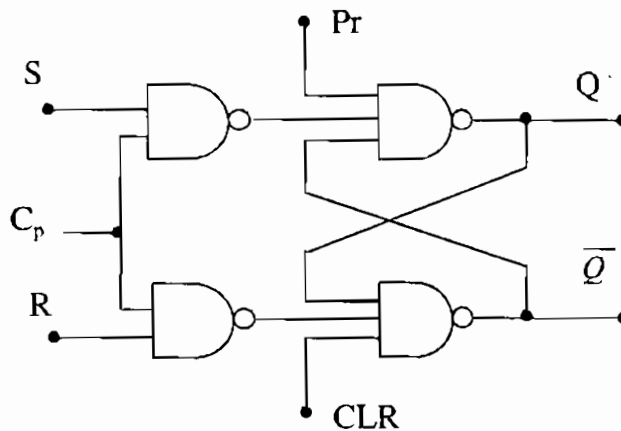
Hình 6.5: RS-FF hoạt động không đồng bộ dùng NAND

2.2. Loại RS- FF hoạt động đồng bộ



Hình 6.6a: RS-FF hoạt động đồng bộ

Hình 6.6b: RS-FF có đầu vào hoạt động không đồng bộ, ưu tiên



Hình 6.6c: RS-FF đồng bộ và có đầu vào không đồng bộ ưu tiên

Ở loại RS - FF hoạt động đồng bộ, ngoài hai đầu vào R và S còn có đầu vào xung C_p . Hình 6.6.b vẽ loại RS - FF hoạt động đồng bộ và còn có khả năng hoạt động không đồng bộ ưu tiên. Việc hoạt động không đồng bộ ưu tiên được thực hiện bởi 2 đầu vào Pr và CLR. Gọi là hoạt động không đồng bộ vì Pr và CLR đặt được các trạng thái của Q mà không cần xung nhịp. Pr đặt Q lên logic 1 còn CLR đặt Q về logic 0. Ưu tiên ở đây được giải thích như sau: Nếu đồng

thời có tín hiệu điều khiển đồng bộ (R, S, C_p) và tín hiệu Pr hoặc CLR thì việc lật trạng thái FF là do Pr hoặc CLR quyết định, còn lúc đó các tín hiệu đồng bộ không có tác dụng.

Bảng 6.3.a. Bảng trạng thái của RS- FF

Dòng	S	R	Q	Q'
1	0	0	0	0
2	0	0	1	1
3	0	1	0	0
4	0	1	1	0
5	1	0	0	1
6	1	0	1	1
7	1	1	0	x
8	1	1	1	x

Bảng 6.3.b: Bảng chức năng rút gọn của RS - FF

S	R	Q'
0	0	Q
0	1	0
1	0	1
1	1	X

3. Nguyên lý hoạt động

Bảng trạng thái của RS-FF được trình bày trong bảng 6.3a. Ba cột đầu S, R và Q là giá trị của các tín hiệu vào điều khiển (S,R) và trạng thái của FF (Q) tại thời điểm t. Các giá trị ở các hàng của 3 cột này biểu diễn tất cả các tổ hợp giá trị có thể được của S, R và Q. Cột thứ tư là trạng thái sẽ chuyển biến tới (ký hiệu là Q') của FF sau thời gian quá độ là δt : Q' là trạng thái ra của FF ở thời điểm tiếp theo t + δt .

Rút gọn bảng 6.3.a, ta có hoạt động của RS-FF loại không đồng bộ như sau:

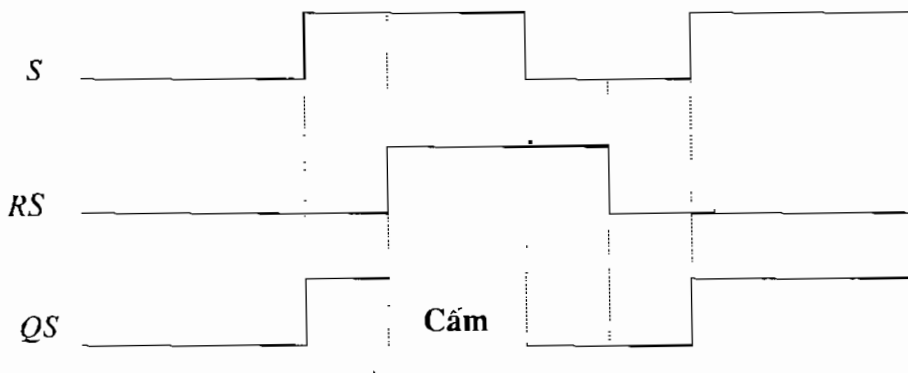
- Khi $S = R = 0$, FF giữ nguyên trạng thái cũ $Q' = Q$.
- Khi $S = 0, R = 1$, đầu vào xoá (Reset) có tín hiệu, FF sẽ chuyển đến trạng thái 0, nghĩa là $Q' = 0$ dù rằng trước đó FF ở trạng thái 0 hay 1.

- Khi $S = 1$, $R = 0$, đầu vào thiết lập "1" (Set) có tín hiệu, FF sẽ chuyển đến trạng thái 1, nghĩa là $Q' = 1$.

- Tổ hợp tín hiệu vào $R = S = 1$ là tổ hợp tín hiệu vào cấm của RS-FF. Khi đó FF nhận được đồng thời 2 tín hiệu điều khiển thiết lập "1" ($S = 1$) và xoá ($R = 1$) trạng thái của FF sẽ không xác định.

Trạng thái rút gọn của RS-FF được biểu diễn trên bảng 6.3.b.

Biểu đồ thời gian mô tả hoạt động của RS - FF loại không đồng bộ được vẽ ở hình 6.7.



Hình 6.7: Hoạt động của RS - FF loại không đồng bộ

Hoạt động của RS - FF loại đồng bộ cũng gần tương tự như của RS - FF không đồng bộ nhưng chỉ có điều khác là cần có sự tham gia của xung nhịp với cực tính thích hợp (dương hoặc âm). Có thể tóm tắt như dưới đây.

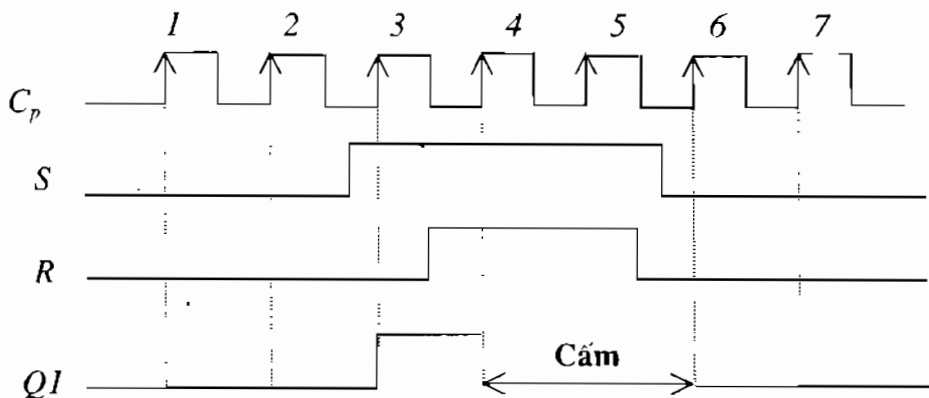
Khi $R=0$, $S = 0$, C_p thích hợp ($\uparrow$ hoặc $\downarrow$) Q vẫn giữ nguyên trạng thái.

Khi $R=0$, $S = 1$, C_p thích hợp ($\uparrow$ hoặc $\downarrow$) $Q = "1"$

Khi $R=1$, $S = 0$, C_p thích hợp ($\uparrow$ hoặc $\downarrow$) $Q = "0"$

Khi $R=1$, $S=1$, C_p thích hợp ($\uparrow$ hoặc $\downarrow$) Không xác định được trạng thái của Q.

Mô tả hoạt động của RS - FF loại đồng bộ được thể hiện ở hình 6.8



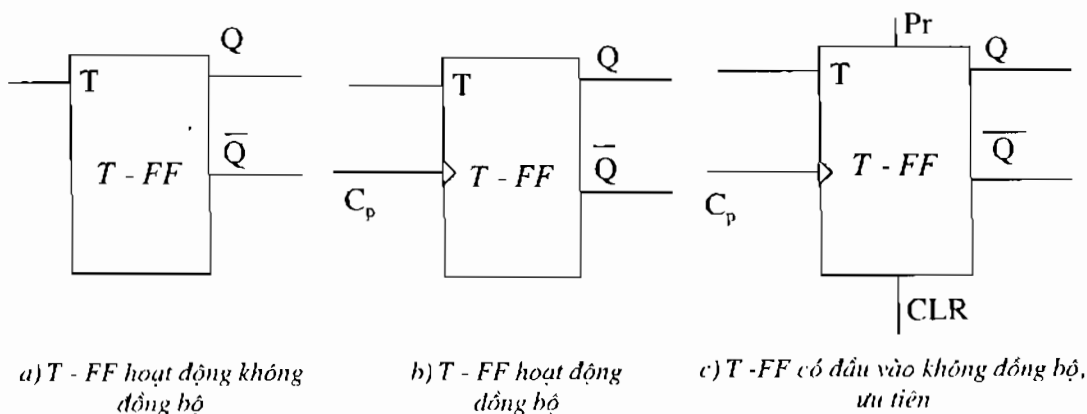
Hình 6.8: Hoạt động của RS - FF loại đồng bộ

Qua biểu đồ thời gian ở hình 6.11 chúng ta thấy việc lật trạng thái đầu ra Q diễn ra vào thời điểm có sườn dương ($\uparrow$) của xung C_p (khi R, S và trạng thái đầu ra Q trước đó đã phù hợp).

III. T - FF

1. Định nghĩa

Một cách tổng quát, T-FF là loại FF có một đầu vào điều khiển T. Sơ đồ ký hiệu của nó được biểu diễn trên hình 6.9.a.



Hình 6.9: Sơ đồ ký hiệu T-FF

Phân loại chi tiết theo nguyên tắc làm việc, T-FF bao gồm các loại:

- Loại hoạt động không đồng bộ, ký hiệu như hình 6.9a
- Loại hoạt động đồng bộ, ký hiệu như hình 6.9b

- Loại T-FF đồng bộ có thêm các đầu vào hoạt động không đồng bộ, ưu tiên (hình 6.9c)

2. Nguyên lý hoạt động

* Bảng chân lý của T-FF được cho ở bảng 6.4a. Từ bảng này có thể đơn giản để nhận được bảng chức năng rút gọn 6.4b.

Bảng 6.4a: Bảng chân lý của T-FF Bảng 6.4b: Bảng chức năng rút gọn

T	Q	Q'
0	0	0
0	1	1
1	0	1
1	1	0

T	Q'
0	Q
1	$\overline{Q}$

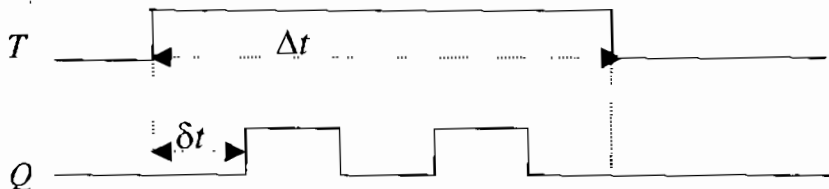
Nhận thấy rằng khi $T=0$, FF giữ nguyên trạng thái cũ, $Q' = Q$. Còn khi $T = 1$, FF lật trạng thái (Toggle). Cũng chính vì thế mà FF này có tên là T-FF.

Phương trình đặc trưng T-FF được biểu diễn ở dạng:

$$Q' = \overline{T}Q + T\overline{Q} = T \oplus Q$$

Giả thiết rằng tín hiệu vào T tồn tại trong thời gian Δt và δt là thời gian trễ của mạch. Giả sử ban đầu của FF ở trạng thái 0, thì sau δt , FF sẽ lật lên trạng thái 1. Nếu tín hiệu vào T vẫn chưa kết thúc (tức là $\Delta t > \delta t$) thì mạch lại sẽ tiếp tục lật về trạng thái 0.

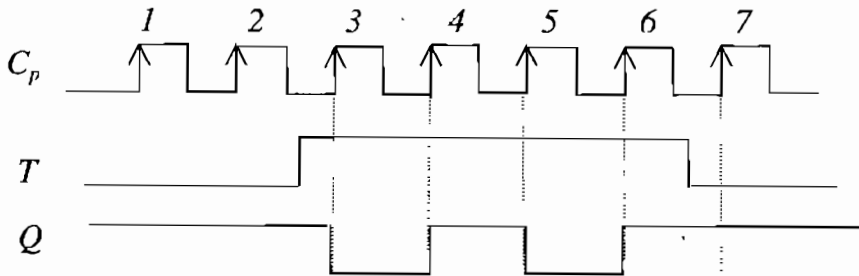
Quá trình cứ tiếp tục như vậy cho đến khi đầu vào T trở về 0. Mạch ở trạng thái dao động. Khi kết thúc xung điều khiển T, ta không biết đích xác mạch ở trạng thái nào. Giảm đồ dạng sóng của mạch được vẽ ở hình 6.10. Trong thực tế $\Delta t \gg \delta t$ do vậy mạch luôn luôn ở trạng thái dao động khi $T \approx 1$. T - FF là loại FF chỉ có thể làm việc ở chế độ đồng bộ.



Hình 6.10: Biểu đồ thời gian của T - FF loại không đồng bộ

* Đối với loại T-FF hoạt động đồng bộ, sơ đồ ký hiệu như hình 6.9b, khi $T = 0$, có xung nhịp với cực tính thích hợp cũng không làm thay đổi trạng thái

đầu ra của FF. Khi $T=1$ logic, nếu có xung nhịp đồng bộ FF sẽ lật trạng thái theo sườn xung nhịp tích cực. Biểu đồ thời gian mô tả hoạt động của T - FF đồng bộ theo hình 6.11.



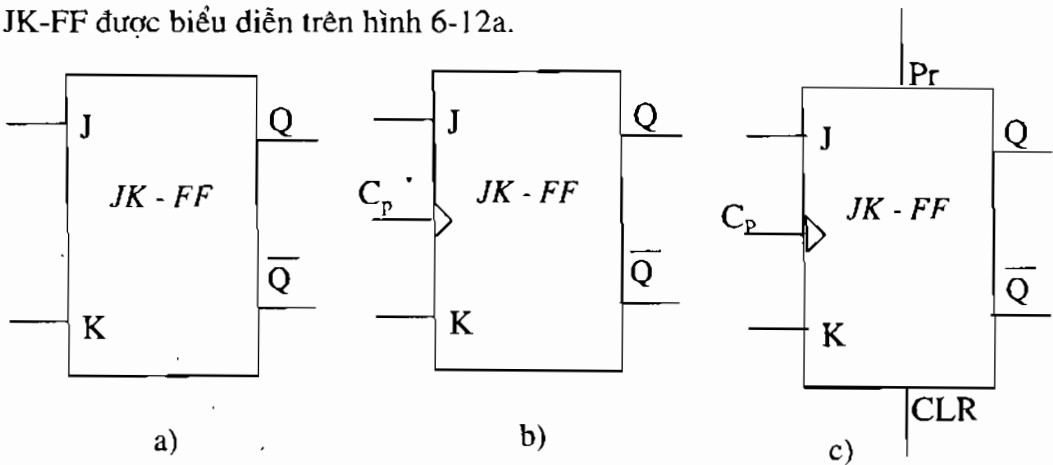
Hình 6.11: Biểu đồ thời gian mô tả hoạt động của T - FF loại đồng bộ

* Hoạt động của loại T-FF có đầu vào không đồng bộ và ưu tiên (hình 6.9c): Nếu 2 đầu vào Pr và CLR không ở mức tích cực, T - FF hoạt động theo các đầu vào đồng bộ T và C_p . Khi Pr và CLR ở mức tích cực, quyền điều khiển FF thuộc về hai đầu vào này giống như ở các loại FF khác đã xét.

IV. JK - FF

1. Định nghĩa

JK-FF là loại FF có 2 đầu vào điều khiển J, K. Sơ đồ ký hiệu tổng quát của JK-FF được biểu diễn trên hình 6-12a.



Hình 6.12: Sơ đồ ký hiệu các loại JK - FF

2. Phân loại JK - FF

Theo nguyên tắc làm việc, người ta chia JK - FF ra làm 2 loại cơ bản:

- JK - FF hoạt động không đồng bộ, hình 6.12a.
- JK - FF hoạt động đồng bộ, hình 6.12b.

Cũng như RS - FF, JK - FF còn có loại được cấu tạo ngoài 3 đầu vào điều khiển đồng bộ còn có thêm 2 đầu vào hoạt động không đồng bộ, ưu tiên (Pr, CLR), sơ đồ ký hiệu ở hình 6.12c.

3. Nguyên lý hoạt động

Bảng chân lý của JK-FF cho ở bảng 6.5a và từ đó suy ra bảng chức năng rút gọn hình 6.5b.

Bảng 6.5a: Bảng chân lý của JK- FF

Dòng	J	K	Q	Q
1	0	0	0	0
2	0	0	1	1
3	0	1	0	0
4	0	1	1	0
5	1	0	0	1
6	1	0	1	1
7	1	1	0	1
8	1	1	1	0

Bảng 6.5b: Bảng chức năng rút gọn

K	Q	Q
0	0	Q
0	1	0
1	0	1
1	1	$\bar{Q}$

Từ bảng 6.5b, nhận thấy rằng:

- Khi JK = 00, FF giữ nguyên trạng thái cũ: $(Q' = Q)$
- Khi JK = 01, FF luôn luôn chuyển đến trạng thái 0: $(Q' = 0)$
- Khi JK = 10, FF luôn luôn chuyển đến trạng thái 1: $(Q' = 1)$
- Khi JK = 11, FF luôn luôn lật trạng thái: $(Q' = \bar{Q})$

Nếu so sánh với chức năng của RS - FF ta nhận thấy có sự tương ứng giữa 2 FF này, cụ thể K tương ứng với R, J tương ứng với S. Tuy nhiên, JK - FF khác so với RS - FF ở chỗ không có tổ hợp tín hiệu vào cấm, cụ thể khi JK = 11, FF luôn luôn lật trạng thái trong khi đối với RS - FF: SR = 11 là tổ hợp tín hiệu vào cấm.

Phương trình đặc trưng của JK - FF có dạng:

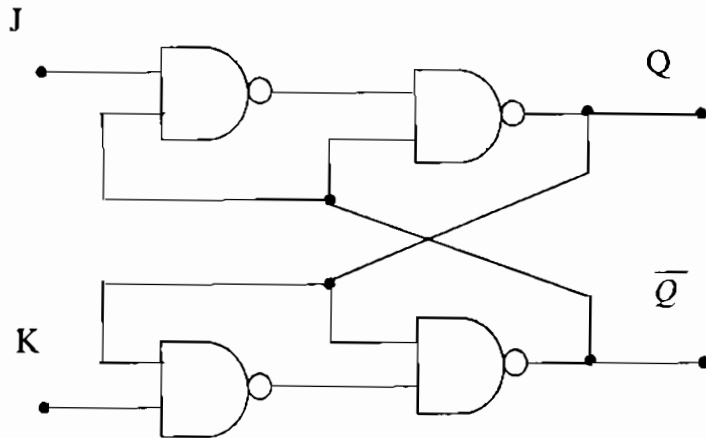
$$Q' = J\bar{Q} + \bar{K}Q.$$

Viết lại phương trình này dưới dạng chỉ dùng các hàm NAND:

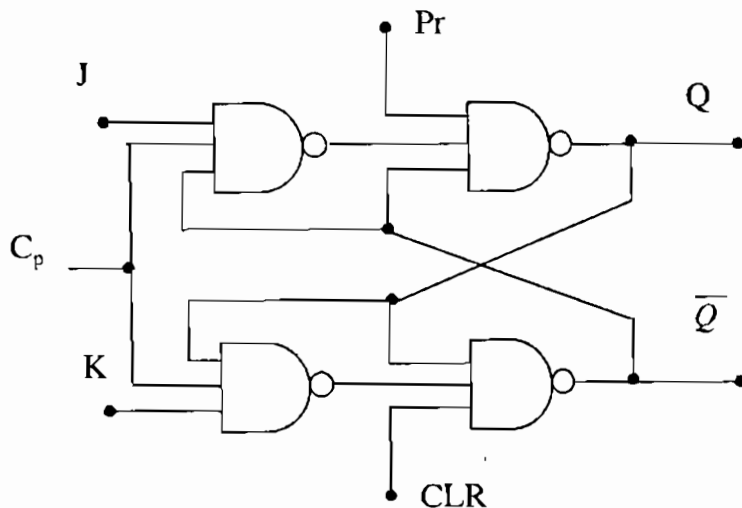
$$Q' = \overline{\overline{J\bar{Q}} \cdot \overline{\bar{K}Q}} = \overline{\overline{J\bar{Q}} \cdot \overline{\bar{K}Q}}$$

$$\bar{Q}' = \overline{\overline{J\bar{Q}} + \bar{K}Q} = \overline{\overline{J\bar{Q}} \cdot \bar{K}Q}$$

Từ đó xây dựng được sơ đồ cho JK - FF như hình 6.13a.



a) Loại hoạt động không đồng bộ

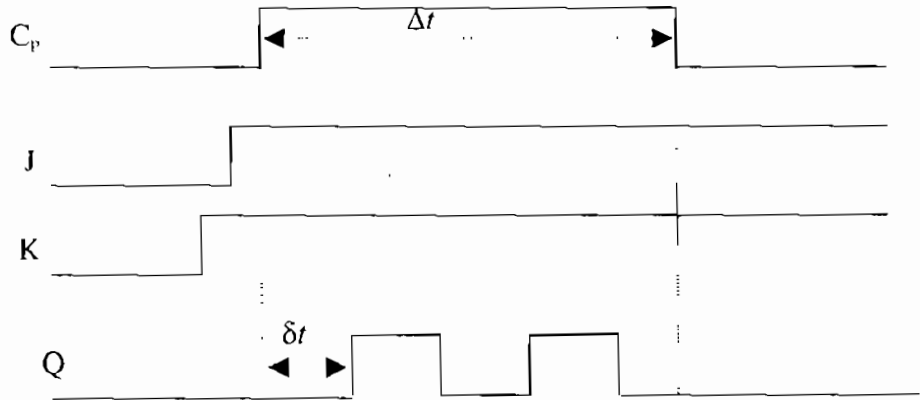


b) Loại hoạt động đồng bộ

Hình 6.13: Sơ đồ JK - FF dùng NAND

Như đã chú ý ở đầu chương, JK-FF chỉ có thể làm việc ở chế độ đồng bộ. Thật vậy, khi $JK = 11$ mạch ở trạng thái dao động. Sơ đồ đầy đủ của JK-FF đồng bộ với các đầu vào xoá, thiết lập (Clear, Preset), xây dựng bằng cổng NAND được biểu diễn ở hình 6.13b.

Đối với mạch cho ở hình 6.13b, các tín hiệu điều khiển J, K chỉ tác động đến mạch khi C_p ở mức logic cao "H", 2 tín hiệu xoá (Clear) và thiết lập (Preset) là những tín hiệu không đồng bộ, mức tích cực thấp "L". Sự hồi tiếp từ đầu ra về đầu vào sẽ làm cho mạch dao động khi xung đồng bộ C_p và các tín hiệu điều khiển J, K ở mức cao.



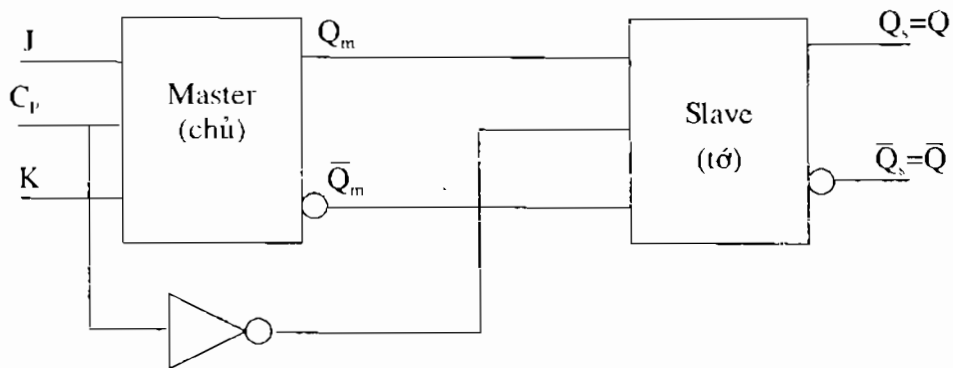
Hình 6.14: Hiện tượng đua vòng quanh của JK - FF

Hiện tượng này được gọi là hiện tượng đua vòng quanh (race around) và có thể gây nên chuyển biến sai nhầm của mạch.

Nếu gọi Δt là khoảng thời gian tồn tại xung C_p , δt là thời gian quá độ của mạch, giản đồ dạng sóng miêu tả hiện tượng đua vòng quanh khi $C_p = "H"$, $J = K = "H"$ cho ở hình 6.14.

Để khắc phục hiện tượng này dùng JK-FF loại chủ - tớ (Master - Slave). Sơ đồ khối được biểu diễn ở hình 6.15

JK-FF loại chủ tớ (Master-Slave) gồm 2 FF mắc liên tiếp: FF-Master và FF - Slave, đầu vào của FF-Master là đầu vào của mạch, còn đầu ra của FF-Slave là đầu ra của mạch.



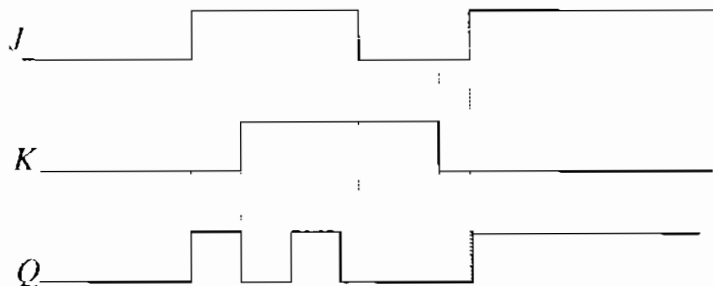
Hình 6.15: Sơ đồ khối JK - FF chủ tớ

Trong hình 6.15, FF Master làm việc khi xung đồng bộ C_p ở mức cao "H", còn FF Slave làm việc khi C_p ở mức thấp "L". Khi xung nhịp ở "H" FF Master sẽ hoạt động theo sự điều khiển của tín hiệu điều khiển ở đầu vào, FF Slave không hoạt động. Khi xung nhịp ở mức "L", ngược lại FF Master không hoạt động, còn FF Slave sẽ thay đổi trạng thái theo sự điều khiển của tín hiệu vào của nó cũng chính là đầu ra của FF Master. Như vậy, khi xung nhịp "H" thông tin ở đầu vào được ghi nhận vào FF Master, còn khi xung nhịp "L" thông tin được đưa ra ở đầu ra của FF Slave cũng chính là đầu ra của mạch. Chính vì cấu tạo như vậy nên FF loại chủ tớ khắc phục được hiện tượng đua vòng quanh.

* Tóm tắt hoạt động của JK - FF loại không đồng bộ:

- Khi JK = 00 đầu ra Q giữ nguyên trạng thái trước đó của nó.
- Khi JK = 01 đầu ra Q được đặt về bằng "0"
- Khi JK = 10 đầu ra Q được đặt về = "1"
- Khi JK = 11 đầu ra Q lật trạng thái (theo hiện tượng đua vòng quanh).

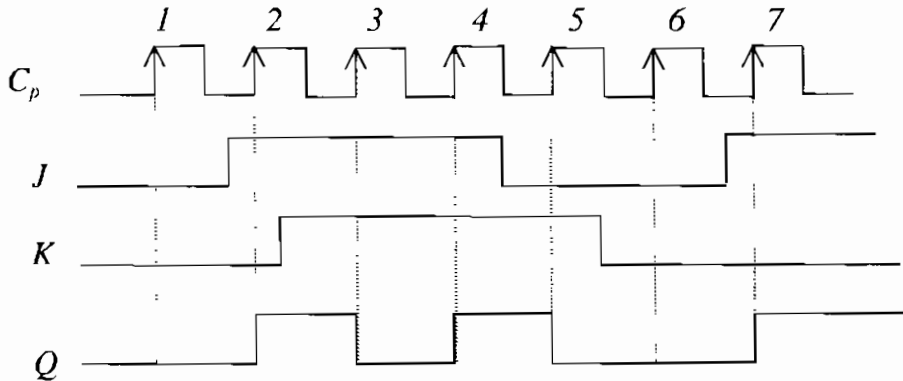
Hình 6.16 là biểu đồ thời gian mô tả hoạt động của JK - FF loại không đồng bộ.



Hình 6.16: Biểu đồ thời gian mô tả hoạt động của JK - FF loại không đồng bộ

* Tóm tắt hoạt động của JK - FF loại đồng bộ (có xung nhịp C_p).

- Khi $J, K = 0, 0$ và có cực tính xung nhịp thích hợp, đầu ra Q giữ nguyên trạng thái trước đó của nó.
- Khi $J, K = 0, 1$ và có cực tính xung nhịp thích hợp, đầu ra Q được đặt về bằng "0"



Hình 6.17: Biểu đồ thời gian mô tả hoạt động của JK - FF loại đồng bộ

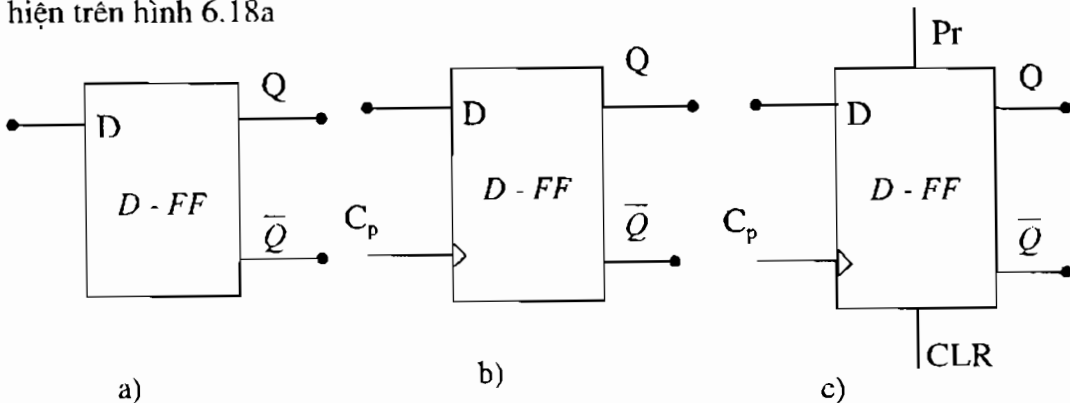
- Khi $J, K = 1, 0$ và có cực tính xung nhịp thích hợp, đầu ra Q được đặt về = "1"
- Khi $J, K = 1, 1$ và có cực tính xung nhịp thích hợp, đầu ra Q lật trạng thái theo xung nhịp.

Hình 6.17 là biểu đồ thời gian mô tả hoạt động của JK - FF loại đồng bộ.

V. D - FF

1. Định nghĩa

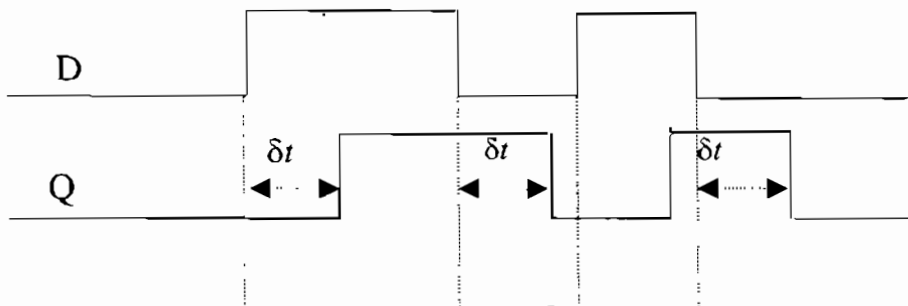
D-FF là loại FF có một đầu vào điều khiển D. Sơ đồ ký hiệu tổng quát thể hiện trên hình 6.18a



Hình 6.18: Sơ đồ ký hiệu các loại D-FF

khoảng thời gian δt . Chính vì vậy, FF này có tên là D - FF (D là viết tắt của Delay - trễ).

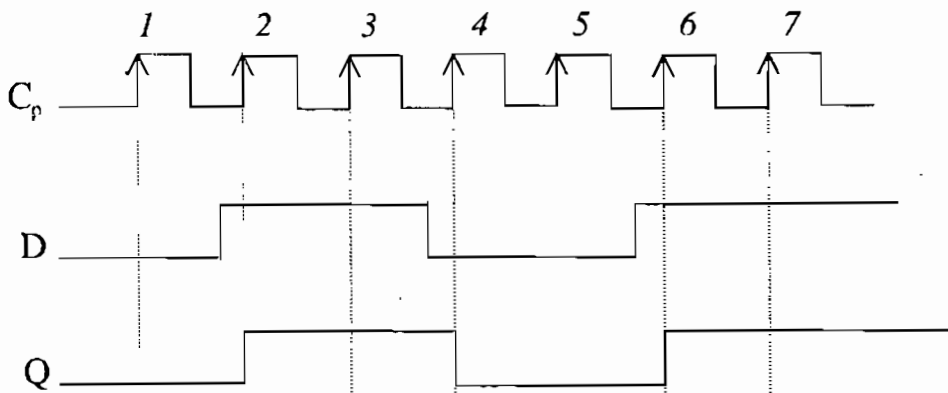
Hình 6.20 là biểu đồ thời gian miêu tả hoạt động của D-FF không đồng bộ.



Hình 6. 20: Biểu đồ thời gian miêu tả hoạt động của D-FF loại không đồng bộ

Khi $D = 0$, Q cũng đang ở trạng thái logic 0. Khi tín hiệu đưa vào D thay đổi, Q không thay đổi trạng thái ngay mà có một thời gian trễ δt .

* Đối với loại D - FF hoạt động đồng bộ, khi có tín hiệu điều khiển đặt vào đầu D , đầu ra nhận được tín hiệu này tại thời điểm tích cực của xung nhịp C_p . Hình 6.21 miêu tả hoạt động của của FF loại này.

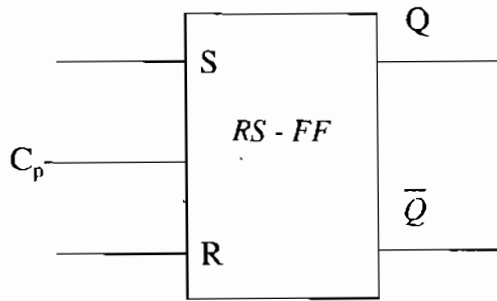


Hình 6.21: Biểu đồ thời gian miêu tả hoạt động của D-FF đồng

VI. MẠCH CHỐT

Các FF có thể làm việc như một mạch chốt (Latch).

Trên hình 6.22 là RS-FF đồng bộ. Khi C_p ở mức "H", đầu ra Q của FF có giá trị theo đúng bảng chân lý của FF. Khi C_p ở mức "L", đầu ra Q của FF giữ không đổi, nghĩa là trạng thái của FF bị chốt lại (xem bảng chức năng 6.7).

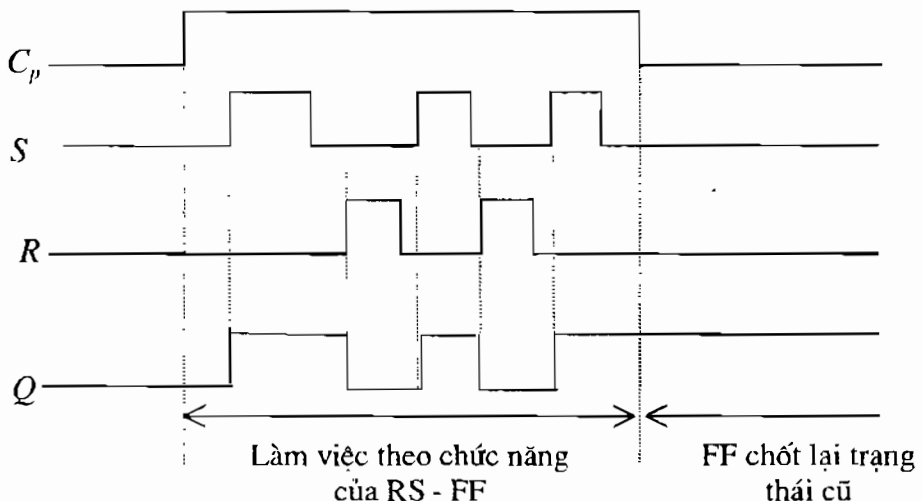


Hình 6.22: Sơ đồ mạch chốt dùng RS - FF

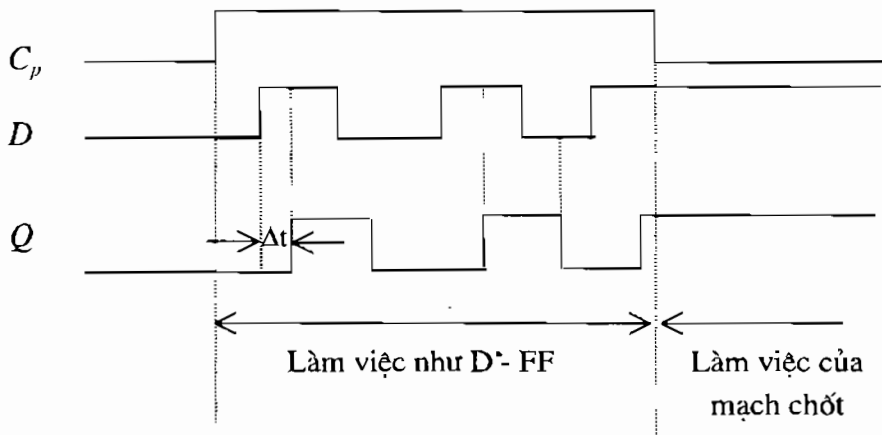
Tương tự giản đồ thời gian cho D-FF làm việc như một mạch chốt được biểu diễn ở hình 6.24.

Bảng 6.7: Bảng chức năng của RS - FF với C_p .

C_p	S	R	Q'	Giải thích
0	0	0	Q	$Q' = Q$, FF bị chốt
0	0	1	Q	
0	1	0	Q	
1	0	0	Q	FF hoạt động theo đúng chức năng.
1	0	1	0	
1	1	0	1	



Hình 6.23: Biểu đồ miêu tả hoạt động của mạch chốt tín hiệu



Hình 6.24: Sơ đồ và biểu đồ thời gian mô tả hoạt động của mạch chốt dùng D-FF.

Câu hỏi và bài tập

1. Nêu định nghĩa, phân loại và ứng dụng của các FF.
2. Thế nào là RS-FF? Vẽ sơ đồ ký hiệu các loại RS - FF.
3. Phân tích hoạt động của RS - FF loại không đồng bộ và đồng bộ, vẽ biểu đồ thời gian mô tả hoạt động của RS - FF đồng bộ với C_p có cực tính âm.
4. Thế nào là D-FF? Vẽ sơ đồ ký hiệu các loại D - FF.
5. Phân tích hoạt động của D - FF loại không đồng bộ và đồng bộ.
6. Thế nào là JK-FF? Vẽ sơ đồ ký hiệu các loại JK - FF.
7. Phân tích hoạt động của JK - FF loại không đồng bộ và đồng bộ.
8. Thế nào là T-FF? Vẽ sơ đồ ký hiệu các loại T - FF.
9. Phân tích hoạt động của T - FF loại không đồng bộ và đồng bộ.
10. Trình bày về mạch chốt.

Chương 7

BỘ ĐẾM

Mục tiêu

- Phân tích được đặc điểm của các loại bộ đếm và nguyên lý hoạt động của nó.
- Biết vận dụng kiến thức của bộ đếm để phân tích kiến thức của các môn học chuyên ngành.
- Hình thành được khả năng tư duy logic, sáng tạo trong công việc chuyên môn.

Nội dung trọng tâm

- Định nghĩa, phân loại và công dụng của bộ đếm.
- Phân tích đặc điểm của sơ đồ và nguyên lý làm việc của các loại bộ đếm đồng bộ, không đồng bộ theo các hướng đếm khác nhau và dung lượng đếm khác nhau.
- Vẽ biểu đồ thời gian mô tả hoạt động của các loại bộ đếm cơ bản.

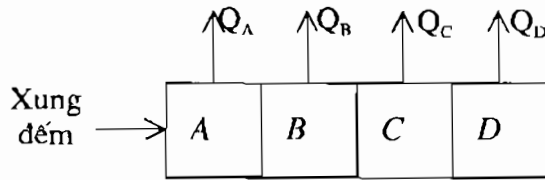
I. KHÁI NIỆM VỀ BỘ ĐẾM

1. Định nghĩa

Đếm là khả năng nhớ được số xung đầu vào. Mạch điện thực hiện được thao tác đếm gọi là bộ đếm.

Mạch đếm là mạch dây đơn giản, cũng như các mạch dây khác, mạch đếm được xây dựng từ các phần tử nhớ là các Flip - Flop và các phần tử tổ hợp (hình 7.1). Nó gồm một đầu vào để nhận các xung đếm, n đầu ra, chúng chính là đầu ra của các FF tham gia cấu tạo nên bộ đếm. Ngoài ra, rất nhiều bộ đếm còn có n đầu vào số liệu hoặc gọi là các đầu vào đặt trước, dùng để đưa một nội dung cho trước vào trong bộ đếm. Một số đầu điều khiển để điều khiển sự làm việc của mạch. Bộ đếm có thể hoạt động theo sườn lên hay sườn xuống của các xung đếm. Nghĩa là chúng chỉ nhận biết có xung đếm tới trong thời gian tương ứng với sườn lên hoặc sườn xuống của xung.

Để xây dựng bộ đếm, người ta có thể dùng mã nhị phân hoặc các loại mã khác như mã Gray, mã NBCD...

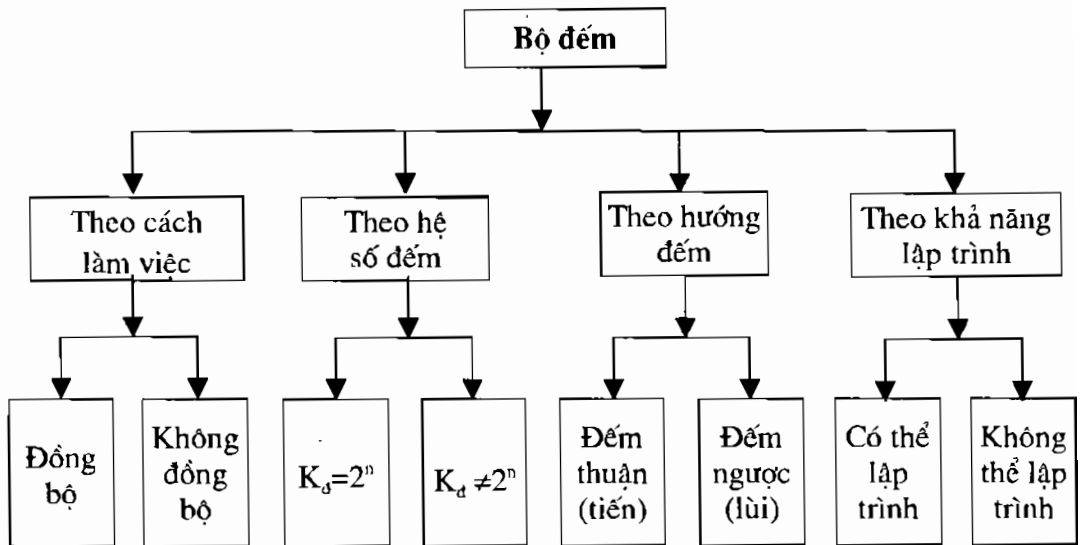


Hình 7.1: Dạng tổng quát của mạch đếm (dùng bốn FF)

Trong tài liệu này chỉ tập trung vào việc phân tích đặc điểm và hoạt động của bộ đếm mà không đề cập tới việc thiết kế bộ đếm.

2. Phân loại bộ đếm

Có nhiều cách để phân loại bộ đếm, sau đây là một vài cách hay được dùng nhất (hình 7.2).



Hình 7.2: Phân loại bộ đếm

2.1. Phân loại theo cách làm việc

- Bộ đếm đồng bộ (Synchronous)
- Bộ đếm không đồng bộ (Asynchronous)

* Bộ đếm đồng bộ (Synchronous counter) là bộ đếm mà các FF dùng để mã hoá cho các trạng thái trong của bộ đếm thay đổi trạng thái cùng một lúc khi có tín hiệu vào đếm và mọi sự chuyển đổi trạng thái (từ S_i sang trạng thái mới S_j với i, j) đều không qua các trạng thái trung gian (hình 7.3.a).

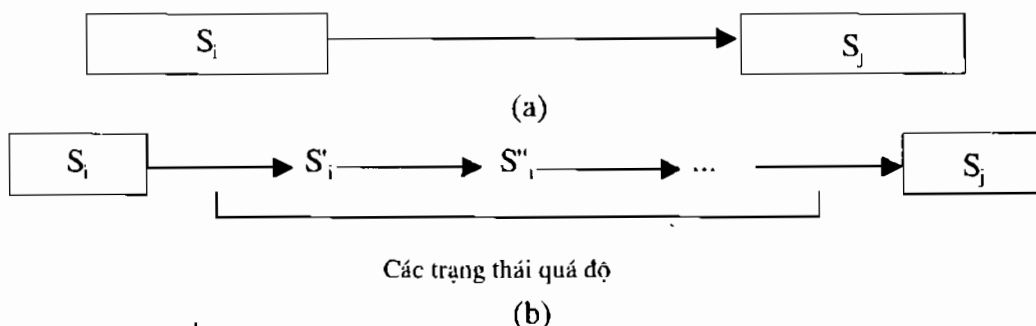
Đặc điểm của bộ đếm này là tín hiệu xung nhịp C_p được đưa đồng thời (song song) vào các Flip - Flop. Bộ đếm loại này cấu trúc phức tạp hơn bộ đếm không đồng bộ.

* Bộ đếm không đồng bộ (Asynchronous counter):

Nếu trong bộ đếm tồn tại ít nhất một cặp chuyển biến trạng thái từ $S_i \Rightarrow S_j$ mà trong đó các FF không thay đổi trạng thái cùng một lúc thì bộ đếm đó được gọi là bộ đếm không đồng bộ.

Đặc điểm của bộ đếm này là tín hiệu xung nhịp C_p không được đưa đồng thời vào các FF, điều này dẫn đến sự lật trạng thái của các FF (nếu có) xảy ra không đồng thời. Ưu điểm của bộ đếm không đồng bộ là có cấu trúc đơn giản.

Ví dụ, sự chuyển từ trạng thái $S_i \Rightarrow S_j$ của bộ đếm không đồng bộ được mô tả như hình 7.3.b.



*Hình 7.3: Đồ hình chuyển đổi trạng thái
của 2 loại bộ đếm đồng bộ và không đồng bộ*

Giả sử ban đầu bộ đếm không đồng bộ đang ở trạng thái cân bằng ổn định S_i , khi có tín hiệu vào đếm, chỉ có một số FF bị tác động, sau đó sự thay đổi trạng thái của các FF này lại dẫn đến sự thay đổi trạng thái của các FF khác. Quá trình cứ tiếp tục cho đến khi bộ đếm ở trạng thái cân bằng ổn định mới S_j . Nghĩa là trong quá trình chuyển đổi giữa hai trạng thái cân bằng ổn định, các FF không thay đổi trạng thái quá độ. Quá trình chuyển qua các trạng thái trung gian trước khi đến trạng thái ổn định mới S_j làm cho tín hiệu trên đầu ra của bộ đếm bị "gợn sóng" (Ripple). Chính vì vậy, bộ đếm không đồng bộ còn được gọi là Ripple Counter.

2.2. Phân biệt theo hệ số (K_d)

- Bộ đếm có hệ số đếm $K_d = 2^n$ (n là số tự nhiên);
- Bộ đếm có hệ số đếm $K_d \neq 2^n$.

* Bộ đếm có hệ số đếm $K_d = 2^n$.

Ví dụ: $K_d = 2, 4, 8, 16...$ Bộ đếm này còn được gọi là bộ đếm có hệ số đếm cực đại hay chiều dài cực đại (Maximum Length), vì khi sử dụng n FF để mã hoá các trạng thái trong bộ đếm thì khả năng mã hoá tối đa là 2^n .

* Bộ đếm có hệ số đếm $K_d \neq 2^n$.

Ví dụ $K_d = 3, 6, 10...$ vẫn sử dụng n FF để mã hoá các trạng thái trong bộ đếm cho nên sẽ có $2^n - K_d$ trạng thái trong không được sử dụng đến. Vì vậy, khi thiết kế bộ đếm này cần phải lưu ý đến các trạng thái không sử dụng đó, tức là cần phải có biện pháp làm cho bộ đếm thoát khỏi các trạng thái đó một cách hợp lý để trở về chu trình đúng mà vẫn phải đảm bảo bộ đếm được thiết kế là đơn giản.

2.3. Phân loại theo hướng đếm

- Bộ đếm thuận hay còn gọi là đếm tiến. (Up Counter);
- Bộ đếm nghịch hay còn gọi là bộ đếm lùi. (Down Counter).

Bộ đếm thuận là bộ đếm mà mỗi khi có tín hiệu vào đếm X_d thì trạng thái trong của bộ đếm tăng lên 1.

Ví dụ: Bộ đếm đang ở trạng thái $S_5 : 101$ khi có một tín hiệu vào đếm X_d nó sẽ chuyển sang trạng thái $S_6 : 110$.

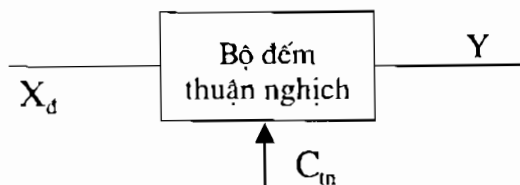
Bộ đếm nghịch:

Là bộ đếm mà mỗi khi có tín hiệu vào đếm X_d thì trạng thái trong của bộ đếm giảm đi 1.

Ví dụ: Ban đầu bộ đếm đang ở trạng thái $S_9 : 1001$ khi có tín hiệu X_d nó sẽ chuyển sang trạng thái $S_8 : 1000$.

Chú ý rằng, khái niệm thuận, nghịch chỉ là tương đối, chủ yếu là do vấn đề mã hoá các trạng thái trong của bộ đếm.

Trong thực tế người ta còn thiết kế bộ đếm thuận nghịch, bộ đếm này vừa có khả năng đếm thuận vừa có khả năng đếm nghịch. Trong trường hợp này phải đưa thêm tín hiệu điều khiển đếm thuận hay đếm nghịch.



Hình 7.4: Sơ đồ khối của bộ đếm thuận nghịch

Sơ đồ khối của bộ đếm thuận nghịch được mô tả ở hình 7.4, trong đó C_{in} là tín hiệu điều khiển việc đếm thuận hay đếm nghịch.

2.4. Phân loại theo khả năng lập trình

- Bộ đếm có khả năng chương trình hoá (Programable Counter);

Bộ đếm có khả năng chương trình hoá là bộ đếm có thể sử dụng với các hệ số đếm khác nhau, tùy thuộc vào tín hiệu điều khiển đưa vào nó. Do vậy, bộ đếm này "mềm dẻo", đa năng hơn các bộ đếm chỉ có 1 hệ số đếm cố định và ngày càng được sử dụng rộng rãi.

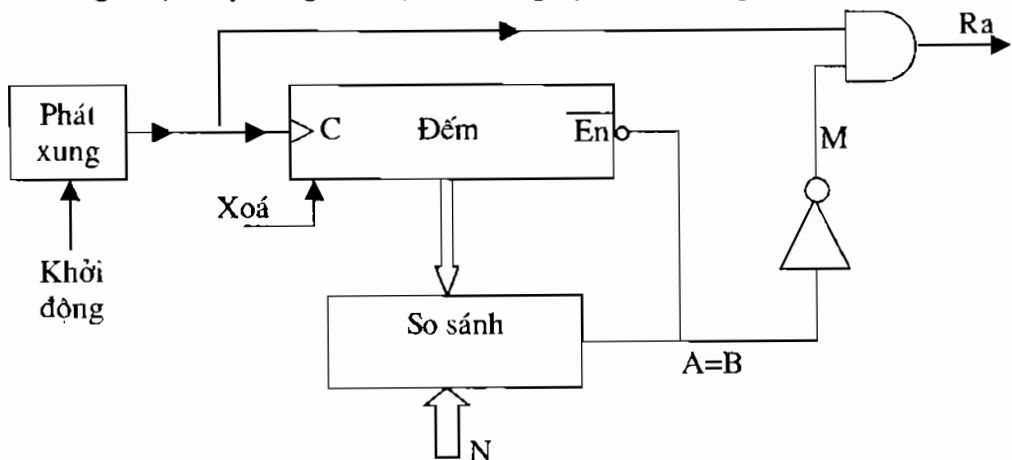
3. Công dụng

Vì đếm là một thao tác cực kỳ quan trọng cho nên bộ đếm được sử dụng vô cùng rộng rãi. Các mạch đếm là thành phần cơ bản của các hệ thống số, chúng được sử dụng để đếm thời gian, chia tần số, điều khiển các mạch khác... Mạch đếm được dùng rất nhiều trong máy tính, trong thông tin, các thiết bị đo chỉ thị số. Bất kỳ hệ thống số hiện đại nào đều hiện diện bộ đếm.

Trong các chương sau bạn sẽ thấy rõ vai trò của bộ đếm. Dưới đây chỉ đơn cử vài ví dụ điển hình.

1. Trong nhiều trường hợp, người ta dùng bộ đếm để tạo lập mạch phát ra một dãy xung có số xung xác định trước (hình 7.5).

Trên hình 7.5, số xung định trước N của dãy xung phát ra được đặt trên một cửa vào của mạch so sánh. Mạch phát xung phát dãy xung vào bộ đếm, đồng thời cũng đưa tới cổng AND để truyền qua đầu ra. Khi số xung phát ra đạt giá trị N định trước thì đầu ra $A = B$ của mạch so sánh nhảy lên 1 logic. Do đó cổng AND đóng, chặn dãy xung tới; bộ đếm cũng bị khoá, không làm việc được.

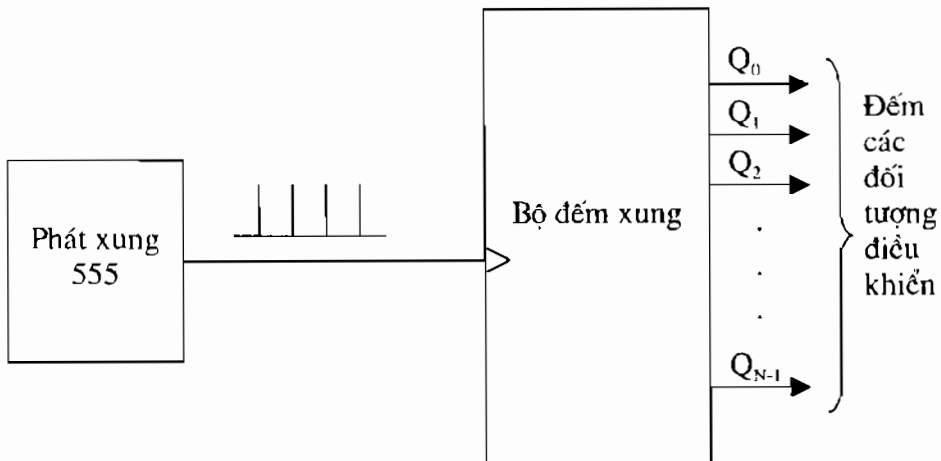


Hình 7.5: Mạch phát dãy xung có số xung xác định trước

2. Một ứng dụng khá phổ biến của bộ đếm là sử dụng nó làm mạch chia tần số của một dãy xung. Ta có thể:

- * Dùng các IC đếm không đồng bộ để tạo thành các mạch chia tần số.
- * Dùng mạch đếm vòng cho việc chia tần số.

3. Người ta còn hay dùng bộ đếm chạy vòng để điều khiển những hoạt động có tính trình tự và chu trình, ví dụ điều khiển ánh sáng chạy của một dãy đèn quảng cáo (hình 7.6). Trên hình vẽ, tín hiệu ở các đầu ra Q_0 và Q_{N-1} sẽ dẫn tới điều khiển sự làm việc của N đối tượng, ví dụ những công tắc các mạch chiếu sáng hàng đèn quảng cáo. Dãy xung nhịp từ mạch phát xung dùng timer 555 làm thay đổi một cách tuần tự mức logic trên các đầu ra bộ đếm vòng, từ Q_0 đến Q_{N-1} và lặp lại sau N xung tới bộ đếm. Kết quả là ta tạo được một ánh sáng chạy vòng phát ra từ hàng đèn quảng cáo.



Hình 7.6: Mạch điều khiển những hoạt động trình tự và chu trình

II. BỘ ĐẾM TIẾN KHÔNG ĐỒNG BỘ MOD TÁM

Là bộ đếm hoạt động không đồng bộ, mỗi khi có xung đếm đưa vào thì trạng thái đầu ra của bộ đếm lại tăng lên. Khả năng đếm tối đa của bộ đếm này là 8 xung. Ta sẽ xét kỹ bộ đếm này theo các nội dung sau.

1. Bảng đếm

Để biểu diễn được số 8 hệ mười cần 3 bit, với 3 bit ký hiệu là C, B, A (tính theo trọng số giảm dần) ta biểu diễn trạng thái của các giá trị từ 0 đến 8 ở hệ nhị phân.

Khi hoạt động, bộ đếm Mod 8 sẽ có trạng thái đầu ra sau khi có xung đếm đưa tới đầu vào theo đúng bảng đếm 7.1.

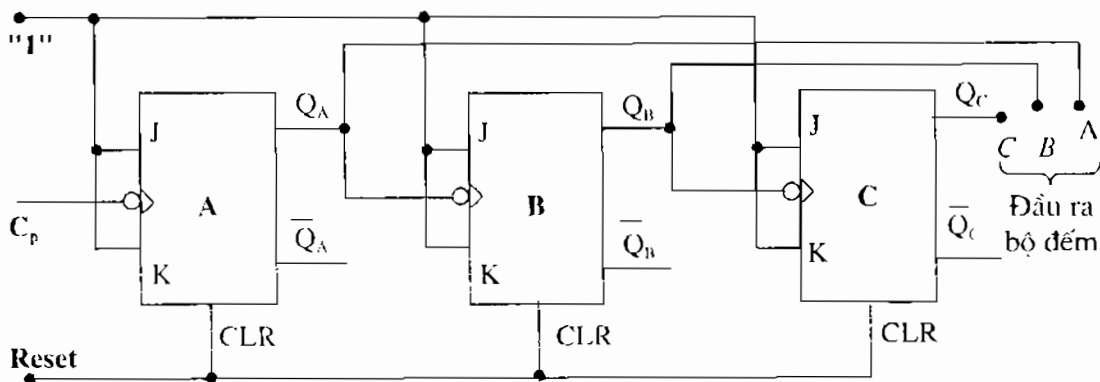
Bảng 7.1: Bảng đếm của bộ đếm Mod 8

Xung đếm	Trạng thái đếm		
	C	B	A
0	0	0	0
1	0	0	1
2	0	1	0
3	0	1	1
4	1	0	0
5	1	0	1
6	1	1	0
7	1	1	1
8	0	0	0

2. Sơ đồ logic

Từ hình 7.7, chúng ta thấy: Bộ đếm được xây dựng từ 3 JK- FF. Trong 3 JK- FF thì JK-FF(A) có trọng số thấp nhất (2^0) và JK-FF(C) có trọng số cao nhất (2^2). Hai đầu J và K của cả 3 FF đều được nối với mức logic 1, với cách nối này khi có xung đếm với cực tính thích hợp đưa vào FF, FF sẽ lật trạng thái. Các FF hoạt động tại thời điểm sườn âm của xung đếm.

Tín hiệu Reset đưa vào đầu CLR của các FF để đặt bộ đếm về trạng thái 0.



Hình 7.7: Sơ đồ bộ đếm không đồng bộ Mod 8

Xung đếm C_p đưa vào FF có trọng số thấp nhất, việc lật trạng thái của 2 FF còn lại phụ thuộc vào trạng thái đầu ra Q của FF đứng trước nó.

Đầu ra của bộ đếm chính là đầu ra Q của các FF. (C là đầu ra của Q_C , B là đầu ra của Q_B và A là đầu ra của Q_A).

3. Nguyên lý hoạt động

* Giả sử trước khi đếm, bộ đếm đang ở trạng thái bất kỳ, tức là đầu ra Q của các FF có thể đang ở một trong hai mức logic 0 hoặc 1.

* Tín hiệu Reset thiết lập trạng thái 0 cho bộ đếm đưa tới đầu vào CLR làm cho đầu ra Q của cả 3 FF đều ở trạng thái 0 logic.

* Xung đếm C_p thứ nhất đưa vào bộ đếm, cụ thể là đưa vào JK-FF(A), do hai đầu J và K của FF này đang nối với mức logic 1 nên tại thời điểm sườn âm của xung đếm thứ nhất đầu ra Q_A của JK-FF(A) lật trạng thái từ “0” lên “1”.

- Khi Q_A chuyển trạng thái từ “0” lên “1” tác động vào JK-FF(B). Mặc dù JK-FF(B) cũng có JK nối với logic 1 nhưng không lật được trạng thái vì sườn xung do Q_A tác động vào là sườn dương nên không phù hợp mức tích cực đối với JK-FF(B). Đầu ra Q_B vẫn giữ ở mức logic 0.

- Vì Q_B không thay đổi trạng thái cho nên JK-FF(C) cũng không thay đổi trạng thái, đầu ra Q_C cũng vẫn ở mức logic 0.

- Như vậy, kết thúc xung đếm thứ nhất ta có $Q_A = “1”$ còn Q_B và Q_C vẫn bằng “0”. ($C=0$, $B=0$ và $A=1$). Điều này phù hợp với bảng đếm của bộ đếm Mod 8 đã nêu ở trên.

* Xung đếm C_p thứ hai đưa vào bộ đếm, vì J và K của FF(A) vẫn đang nối với “1” cho nên FF(A) lại lật trạng thái. Q_A chuyển từ “1” xuống “0”.

- Với việc chuyển trạng thái từ “1” xuống “0” của Q_A , nó tác động vào FF(B) và do FF(B) cũng vẫn đang có J, K nối với “1” nên FF(B) lật được trạng thái. Q_B từ “0” chuyển lên “1”.

- Q_B chuyển trạng thái logic từ “0” lên “1” tác động vào JK-FF(C) nhưng không làm thay đổi trạng thái của JK-FF(C) mặc dù J và K của FF(C) cũng đang được nối với “1”. Bởi vì tín hiệu từ “0” lên “1” tạo ra sườn dương của xung đếm mà các FF như ở sơ đồ logic của bộ đếm đã thể hiện, nó hoạt động với sườn âm của xung đếm đưa vào mỗi FF.

- Kết thúc xung đếm thứ hai bộ đếm có trạng thái 010, ($C=0$, $B=1$ và $A=0$).

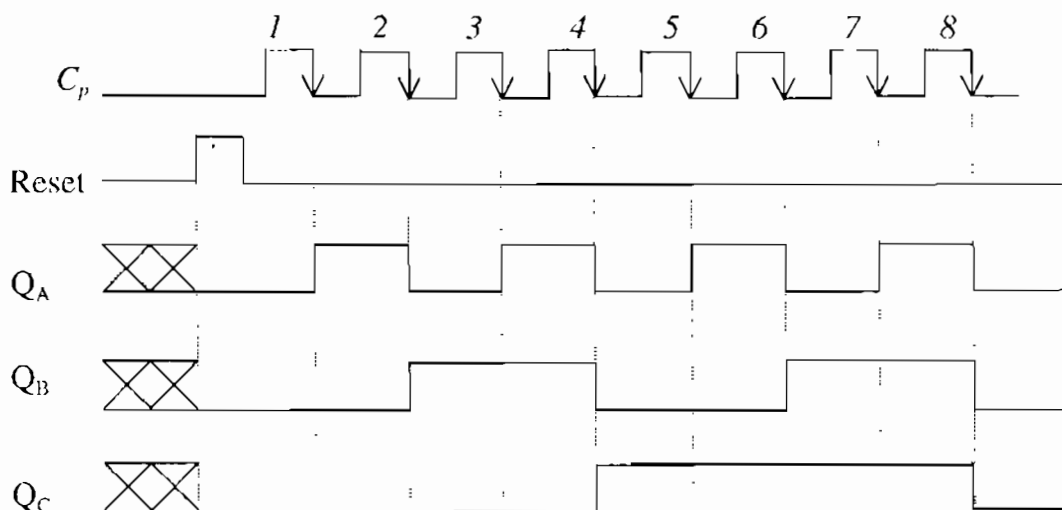
* Xung đếm C_p thứ ba đưa vào bộ đếm làm cho FF(A) lật trạng thái từ “0” lên “1”, điều này dẫn đến FF(B) không lật được trạng thái mà vẫn giữ nguyên ở

mức “1” và như vậy cũng không làm thay đổi trạng thái đầu ra của FF(C). Q_C vẫn bằng “0”.

Sau xung thứ ba, đầu ra bộ đếm có trạng thái 011, ($C=0$, $B=1$ và $A=1$).

* Xung đếm C_p thứ tư làm cho Q_A và Q_B chuyển trạng thái từ “1” về “0”. Khi Q_B chuyển trạng thái từ “1” về “0” làm cho Q_C chuyển từ “0” lên “1”. Điều này có được hoàn toàn giống phân tích ở các trường hợp trên.

* Quá trình cứ như vậy diễn ra cho tới khi hết xung C_p thứ bảy trạng thái bộ đếm là 111, ($C=1$, $B=1$ và $A=1$).



Hình 7.8: Biểu đồ thời gian của bộ đếm Mod 8 không đồng bộ

Hình 7.8 là biểu đồ thời gian miêu tả hoạt động của bộ đếm tiến, Mod 8, hoạt động không đồng bộ. Q_i là đầu ra của các FF trong bộ đếm và cũng chính là đầu ra của bộ đếm.

Qua biểu đồ chúng ta thấy, trước khi đếm trạng thái đầu ra của bộ đếm là bất kỳ. Xung Reset đặt bộ đếm về trạng thái 0 (tất cả các $Q_i = "0"$). Mỗi xung đếm đưa vào Q_A đều thay đổi trạng thái tại thời điểm sườn âm của xung đếm. Q_B thay đổi trạng thái khi Q_A chuyển từ mức logic H xuống mức logic L, (sườn âm của xung). Q_C thay đổi trạng thái do Q_B chuyển từ mức H xuống mức L. Sau 8 xung cả 3 đầu ra Q đều ở mức L, tương đương với trạng thái 0 của bộ đếm. Ở sơ đồ trên đã lý tưởng hoá sự trễ của các tín hiệu, coi thời gian trễ bằng 0. Nhìn vào biểu đồ thời gian chúng ta thấy, cứ 2 xung đếm C_p tạo ra 1 xung ở

đầu ra Q_A , cứ 2 xung Q_A tạo ra 1 xung Q_B và cứ 2 xung Q_B tạo ra một xung Q_C . Điều này thấy rõ một trong ứng dụng của bộ đếm là dùng làm bộ chia tần số.

III. BỘ ĐẾM LÙI KHÔNG ĐỒNG BỘ

1. Bảng đếm

Bộ đếm lùi, hoạt động không đồng bộ là bộ đếm hoạt động theo nguyên tắc không đồng bộ và cứ mỗi xung đếm đưa vào thì trạng thái đầu ra của bộ đếm bị giảm đi một.

Bảng 7.2: Bảng đếm lùi 3 bit

Xung đếm	Trạng thái đếm		
	C	B	A
0	1	1	1
1	1	1	0
2	1	0	1
3	1	0	0
4	0	1	1
5	0	1	0
6	0	0	1
7	0	0	0

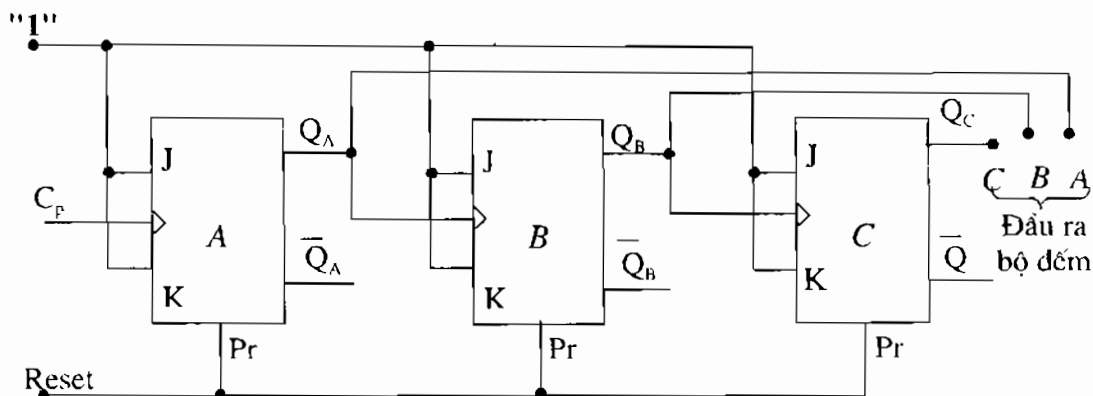
Để tìm hiểu về loại bộ đếm này, ta xét cụ thể vào bộ đếm lùi 3 bit nghĩa là bộ đếm dùng 3 FF và khả năng đếm tối đa của nó là 8 xung. Trạng thái đầu ra của bộ đếm này khi hoạt động được thể hiện qua bảng đếm 7.2.

2. Sơ đồ logic

Hình 7.9 là sơ đồ bộ đếm lùi hoạt động không đồng bộ 3 bit.

Bộ đếm được xây dựng từ 3 JK-FF. Hai đầu J và K của cả 3 FF đều được nối với mức logic 1. Như vậy, khi có xung đưa vào FF ở thời điểm sườn xung thích hợp (đối với sơ đồ hình 7.9, sườn xung thích hợp là sườn dương), FF sẽ lật được trạng thái đầu ra. Xung đếm được đưa vào FF có trọng số thấp nhất trong bộ đếm. Đầu ra của bộ đếm là đầu ra Q của các FF. Ban đầu khi chưa có xung đếm đưa vào, bộ đếm ở trạng thái 111, ($C=1$, $B=1$ và $A=1$).

Tín hiệu Reset đưa tới đầu Preset (Pr) của các FF để đặt bộ đếm về trạng thái ban đầu, (111).



Hình 7.9: Sơ đồ bộ đếm lùi không đồng bộ Mod 8

3. Hoạt động của bộ đếm

Trước khi đếm, trạng thái đầu ra của bộ đếm là 111.

* Khi có xung đếm C_p thứ nhất đưa vào bộ đếm, do J và K của FF (A) đang nối với logic 1 nên tại thời điểm sườn đi lên của C_p , FF(A) lật trạng thái. Q_A đang ở mức logic 1 chuyển thành mức logic 0. Khi Q_A chuyển trạng thái như vậy nó không thể lật được FF (B) vì cực tính xung ở đầu ra của Q_A không thích hợp, mặc dù FF(B) có J, K đang nối với logic 1.

FF(B) không thay đổi trạng thái dẫn đến FF(C) cũng không thay đổi trạng thái. Sau xung đếm thứ nhất, đầu ra của bộ đếm là 110, ($C=1$; $B=1$ và $A=0$) tương đương với dòng thứ hai trong bảng đếm 7.2.

* Xung đếm C_p thứ hai đưa vào bộ đếm, nó tiếp tục làm FF(A) lật trạng thái, ở đầu ra Q_A , tín hiệu chuyển từ mức logic 0 lên logic 1. Việc chuyển mức như vậy làm cho FF(B) lật trạng thái. Ở đầu ra Q_B từ logic 1 chuyển xuống logic 0. Tín hiệu này đưa vào đầu vào xung của FF(C) nhưng không lật được FF(C) vì cực tính xung không phù hợp.

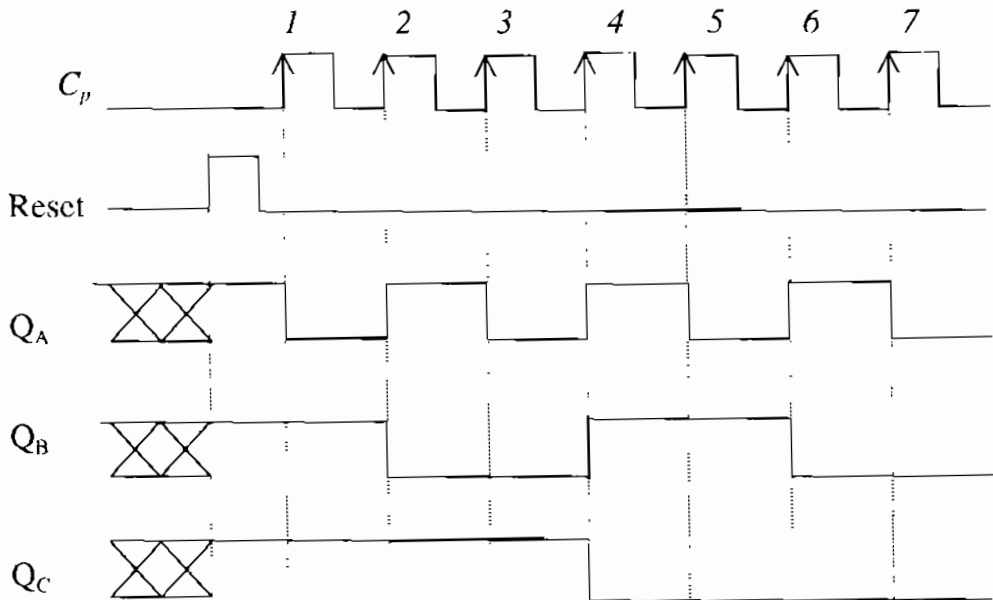
Sau xung đếm thứ hai, trạng thái của bộ đếm là 101 ($C=1$, $B=0$ và $A=1$).

* Quá trình cứ như vậy cho đến hết xung thứ 7 trạng thái đầu ra của bộ đếm là 000, ($C=0$, $B=0$, $A=0$). Nếu có xung đếm thứ 8 đưa vào bộ đếm làm FF(A) lật trạng thái Q_A ở mức logic 0 chuyển lên logic 1, việc chuyển mức này

làm lật FF(B), $Q_B = 1$ logic. Q_B lại tác động tới FF(C) và làm cho Q_C cũng chuyển lên 1. Kết quả là đầu ra bộ đếm ở trạng thái 111.

Hình 7.10 là biểu đồ thời gian mô tả hoạt động của bộ đếm lùi không đồng bộ 3 bit. Trên biểu đồ đã lý tưởng hoá dạng xung cũng như thời gian trễ của các tín hiệu (các xung đều có thời gian trễ = 0, sườn xung có thời gian đột biến rất nhỏ).

Trường hợp muốn có bộ đếm lùi n bit thì cũng có thể ghép n JK - FF theo nguyên tắc bộ đếm lùi hoạt động không đồng bộ 3 bit hình 7.9. Trong trường hợp này, khả năng đếm tối đa của bộ đếm là 2^n xung.



Hình 7.10: Biểu đồ thời gian bộ đếm lùi Mod 8 không đồng bộ

IV. BỘ ĐẾM TIẾN LÙI KHÔNG ĐỒNG BỘ

1. Nguyên tắc chung

Bộ đếm này hoạt động theo nguyên tắc không đồng bộ. Nó vừa có khả năng đếm tiến vừa có khả năng đếm lùi. Điều này phụ thuộc vào tín hiệu điều khiển hướng đếm. Nếu điều khiển theo hướng đếm tiến, mỗi xung đếm C_p đưa vào bộ đếm, trạng thái đầu ra của nó lại tăng lên 1. Nếu điều khiển theo hướng đếm lùi, mỗi xung đếm lại làm trạng thái đầu ra của bộ đếm giảm đi 1.

- Xung đếm C_p thứ hai lại lật được trạng thái của FF(I), vì T của FF(I) vẫn đang nối với mức 1. Xung thứ hai này làm cho Q_I chuyển từ mức 1 xuống mức 0, tạo ra sườn âm của xung. Tín hiệu này qua AND 1 qua OR 1 lật FF(II) từ 0 lên 1. Q_{II} tạo ra sườn xung dương qua cổng AND 3 đưa vào FF(III) nhưng không lật được trạng thái của FF(III) do sườn xung đưa vào không ở mức tích cực.

- Kết thúc xung C_p thứ hai, đầu ra bộ đếm có trạng thái là 010.

- Quá trình cứ như vậy tiếp diễn như hoạt động của bộ đếm tiến hoạt động không đồng bộ có $K_d = 2^n$. Với sơ đồ hình 7.11 khả năng đếm tối đa là 8 xung.

** Hoạt động đếm lùi:*

- D được đặt ở mức logic 0. Điều này làm cho đầu ra của 2 cổng AND 1 và AND 3 luôn ở mức logic 0.

- $D = 0$ qua cổng NOT có mức logic 1 đặt vào đầu vào của 2 cổng AND 2 và AND 4, đầu ra của 2 cổng AND phụ thuộc vào đầu vào còn lại, cụ thể là phụ thuộc vào Q_I và Q_{II} .

Khi xung đếm C_p thứ nhất đưa vào bộ đếm, FF(I) lật trạng thái Q_I từ 1 chuyển xuống 0. $\overline{Q_I}$ chuyển từ 0 lên 1 qua cổng AND 2 không lật được FF(II) vì sườn xung từ Q_I đưa tới không ở mức tích cực.

Sau xung đếm thứ nhất, trạng thái bộ đếm là 110. Xung C_p thứ hai lại lật FF(I), đầu ra Q của FF(I) lại chuyển trạng thái từ 0 lên 1, qua AND 2 và OR 1, FF(II) lật trạng thái, Q_{II} chuyển từ 1 xuống 0, điều này làm FF(III) vẫn chưa lật trạng thái.

Sau khi kết thúc xung đếm thứ 2, trạng thái bộ đếm là 101.

Quá trình cứ như vậy tiếp diễn, cho đến hết xung thứ bảy trạng thái bộ đếm là 000. Xung thứ tám làm bộ đếm lại được đặt ở trạng thái 111.

Có thể dùng các loại FF khác để xây dựng bộ đếm tiến lùi n bit hoạt động không đồng bộ.

4. Giới thiệu IC làm chức năng đếm tiến lùi

4.1. IC 74192, 74193

Hình 7.12 vẽ sơ đồ khối của IC 74192 và IC 74193. Hai IC này có 2 đầu vào đếm UP và DOWN. Nếu xung đếm đưa vào UP (còn đầu DOWN đặt ở giá trị 1 logic) thì bộ đếm sẽ đếm thuận. Ngược lại, nếu xung đếm vào DOWN (còn UP bằng 1 logic), bộ đếm sẽ đếm ngược. Nếu đồng thời có hai dây xung đưa vào UP và DOWN thì mạch sẽ hoạt động theo cách đếm thuận nghịch.

Mạch có đầu CLR (Clear) để xoá nội dung bộ đếm. Khi CLR = 1 logic, bộ đếm bị xoá. Vậy, để mạch có thể đếm được thì phải đặt CLR = 0 logic và LOAD = 1 logic.

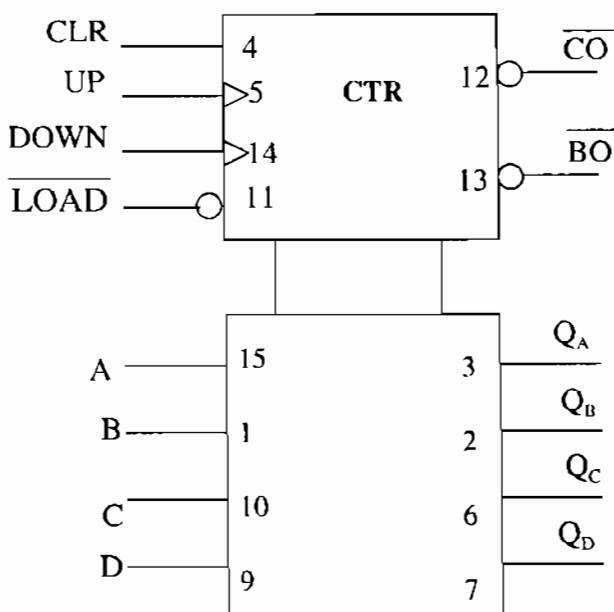
Các đầu vào $\overline{BO}$, $\overline{CO}$ có quan hệ logic với các đầu vào ra khác như sau:

$$\overline{BO} = \overline{\overline{DOWN} \cdot \overline{Q_A} \cdot \overline{Q_B} \cdot \overline{Q_C} \cdot \overline{Q_D}}$$

$$\overline{CO} = \overline{UP \cdot Q_A \cdot Q_D} \text{ (Đối với IC 74192)}$$

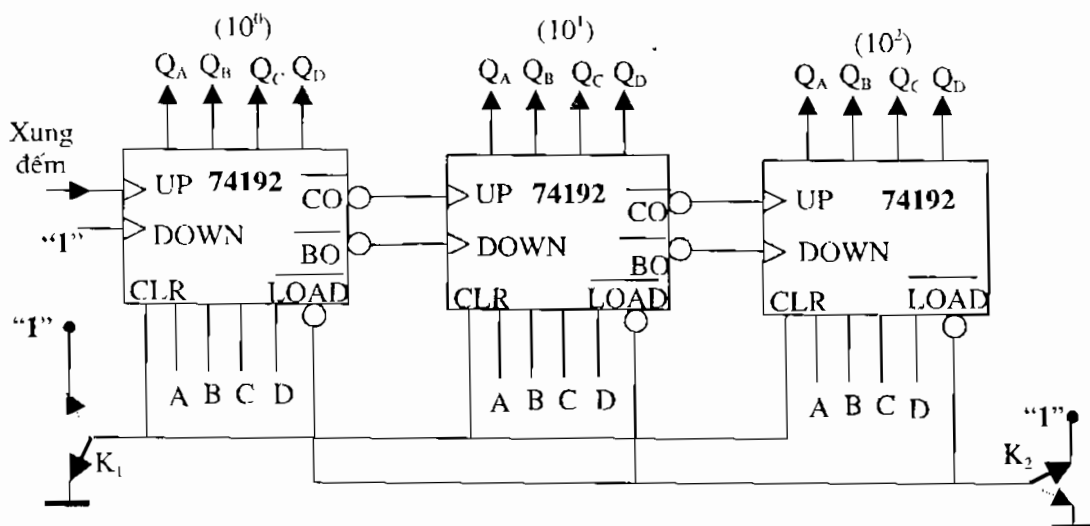
$$\overline{CO} = \overline{UP \cdot Q_A \cdot Q_B \cdot Q_C \cdot Q_D} \text{ (Đối với 74193)}$$

Ta thấy khi đếm ngược thì CO luôn bằng 1 logic (vì UP = 1 logic), còn đầu ra BO chỉ nhảy từ giá trị 1 logic xuống 0 logic khi nội dung bộ đếm giảm xuống số 0 ($Q_A = Q_B = Q_C = Q_D = 0$) và không có xung đếm ở đầu vào DOWN. Lúc đếm thuận thì ngược lại, đầu BO luôn bằng 1 logic, đầu ra CO chỉ nhảy từ 1 logic xuống 0 logic khi bộ đếm đã đạt tới dung lượng của nó $N_{\max}$ ($N_{\max} = 9$ với IC 74192 và =15 với IC 74193) và không có xung ở đầu vào UP.



Hình 7.12 : Sơ đồ khối của các IC 74192 và IC 74193

Hình 7.13 giới thiệu cách nối tăng các IC 74192 và IC 74193. Xung đếm chỉ đi vào UP, đầu DOWN cố định giá trị 1 logic. Nếu đếm ngược thì xung đếm dẫn tới đầu DOWN và UP = 1 logic. Nếu đếm thuận nghịch thì hai dây xung đếm dẫn tới hai đầu vào UP và DOWN. Xem hoạt động của các đầu ra BO, CO ở hình 7.12 trên ta sẽ thuyết minh được hoạt động của mạch hình 7.13



Hình 7.13 : Cách nối tăng các IC 74192, 74193

V. BỘ ĐẾM MOD 10 KHÔNG ĐỒNG BỘ

Đây là bộ đếm thông dụng nhất vì nó có khả năng đếm tối đa là 10 xung, điều này phù hợp với hệ đếm mà con người sử dụng phổ thông.

Bộ đếm thập phân là bộ đếm theo mã nhị - thập phân. Vì tương ứng với nhiều kiểu mã hoá của mã nhị - thập phân, nên cũng có nhiều kiểu bộ đếm thập phân khác nhau. Trong phạm vi của giáo trình này, chúng ta tìm hiểu loại bộ đếm thập phân theo mã 8421.

1. Bảng đếm

Bộ đếm thập phân là loại bộ đếm có $K_d \neq 2^n$, (n là số bit có trong bộ đếm). Ở đây, để đếm đủ 10 xung cần $n = 4$. Với $n = 4$ ta có bảng đếm như bảng 7.3. Quan sát bảng chân lý này ta thấy:

- Với 4 bit D, C, B và A khả năng đếm tối đa là 16 xung.
- A chuyển đổi trạng thái sau mỗi xung đếm.
- B chỉ chuyển đổi trạng thái khi A từ 1 xuống 0.

- C chuyển đổi trạng thái theo B khi B từ 1 xuống 0.
- D chuyển đổi trạng thái khi A từ 1 xuống 0 với điều kiện B và C lúc đó đang = 1.
- Sau xung đếm thứ 10, trạng thái bộ đếm là 1010. Như vậy lúc này D và B = 1 còn C và A = 0. Muốn có bộ đếm mười, sau xung thứ 10 phải chuyển trạng thái đếm về = 0 (cả 4 bit D, C, B, A đều về bằng 0). Dựa vào đặc điểm trạng thái sau xung thứ mười, dùng 2 bit D và B điều khiển đưa bộ đếm về trạng thái 0.

Bảng 7.3. Bảng đếm 4 bit

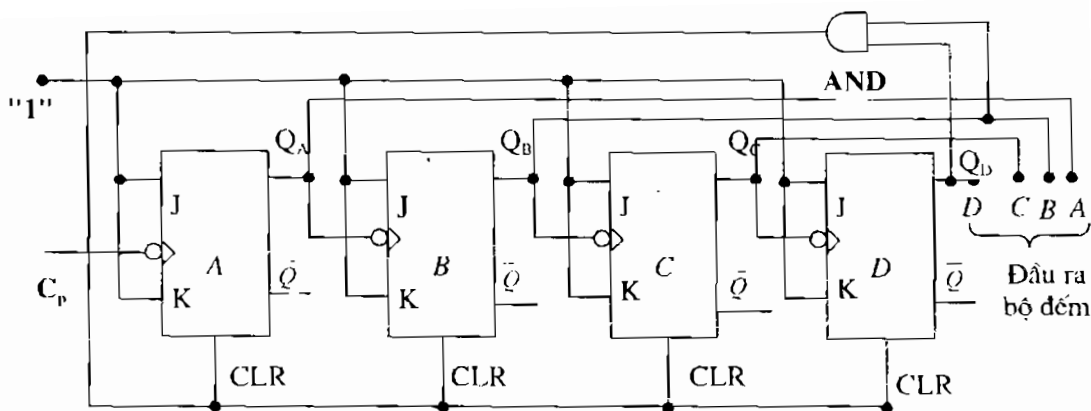
Xung đếm	Trạng thái đếm			
	D	C	B	A
0	0	0	0	0
1	0	0	0	1
2	0	0	1	0
3	0	0	1	1
4	0	1	0	0
5	0	1	0	1
6	0	1	1	0
7	0	1	1	1
8	1	0	0	0
9	1	0	0	1
10	1	0	1	0
11	1	0	1	1
12	1	1	0	0
13	1	1	0	1
14	1	1	1	0
15	1	1	1	1
16	0	0	0	0

Từ các nhận xét trên ta có thể tìm hiểu bộ đếm như hình 7.14.

2. Sơ đồ logic

Có thể dùng các loại FF để xây dựng lên bộ đếm thập phân.

Hình 7.14 vẽ sơ đồ logic của bộ đếm tiến, dung lượng đếm tối đa là 10 xung, hoạt động theo nguyên tắc không đồng bộ, được xây dựng từ loại JK - FF.



Hình 7.14: Sơ đồ bộ đếm tiến, thập phân, hoạt động không đồng bộ

Mạch sử dụng 4 JK-FF loại hoạt động đồng bộ. Cũng như các bộ đếm hoạt động không đồng bộ đã phân tích, ở bộ đếm này, hai đầu J và K của 4 FF đều nối với logic "1".

Xung đếm đưa vào FF có trọng số thấp nhất trong bộ đếm. Từ FF thứ hai trở đi, việc lật trạng thái của các FF phụ thuộc vào FF đứng trước nó. Các FF được lật trạng thái bởi tích cực là sườn âm của xung.

Đầu ra của bộ đếm là đầu ra Q của các FF, (có 4 đầu ra D, C, B, A - xét theo trọng số giảm dần).

Từ đầu ra D và B của bộ đếm, nối vào cổng AND hai đầu vào; từ đầu ra của cổng AND, nối vào đầu CLR của các FF trong bộ đếm để xóa bộ đếm về không.

3. Nguyên lý hoạt động

Giả sử trước khi đếm, bộ đếm đã được đặt ở trạng thái 0000. (Đầu ra bộ đếm bằng 0000).

* Khi có xung đếm C_p thứ nhất đưa vào bộ đếm, FF(A) do có J và K nối ở logic 1 nên nó lật trạng thái tại thời điểm sườn âm của xung đếm. Đầu ra Q_A chuyển từ 0 lên 1.

- Khi Q_A chuyển trạng thái từ "0" lên "1" tác động vào JK-FF(B). Mặc dù JK-FF(B) cũng có J và K nối với logic 1 nhưng không lật được trạng thái vì sườn xung do Q_A tác động vào là sườn dương nên không phù hợp mức tích cực đối với JK-FF(B). Đầu ra Q_B vẫn giữ ở mức logic 0.

- Vì Q_B không thay đổi trạng thái cho nên JK-FF(C) và JK - FF(D) cũng không thay đổi trạng thái, đầu ra Q_C và Q_D vẫn ở mức logic 0.

- Như vậy, kết thúc xung đếm thứ nhất ta có $Q_A = "1"$ còn Q_B , Q_C và Q_D vẫn bằng $"0"$. ($D = 0$, $C = 0$, $B = 0$ và $A = 1$). Điều này phù hợp với bảng đếm 7.3.

* Xung đếm C_p thứ hai đưa vào bộ đếm, vì J và K của FF(A) vẫn đang nối với $"1"$ cho nên FF(A) lại lật trạng thái. Q_A chuyển từ $"1"$ xuống $"0"$.

- Với việc chuyển trạng thái từ $"1"$ xuống $"0"$ của Q_A , nó tác động vào FF(B) và do FF(B) cũng vẫn đang có J, K nối với $"1"$ nên FF(B) lật được trạng thái. Q_B từ $"0"$ chuyển lên $"1"$.

- Q_B chuyển trạng thái logic từ $"0"$ lên $"1"$ tác động vào JK-FF(C) nhưng không làm thay đổi trạng thái của JK-FF(C) mặc dù J và K của FF(C) cũng đang được nối với $"1"$. Bởi vì tín hiệu từ $"0"$ lên $"1"$ tạo ra sườn dương của xung đếm mà các FF như ở sơ đồ logic của bộ đếm đã thể hiện, nó hoạt động với sườn âm của xung đếm đưa vào mỗi FF. FF(C) không thay đổi trạng thái nên FF(D) cũng không thay đổi trạng thái.

- Kết thúc xung đếm thứ hai, bộ đếm có trạng thái 0010, ($D=0$, $C=0$, $B=1$ và $A=0$).

* Xung đếm C_p thứ ba đưa vào bộ đếm làm cho FF(A) lật trạng thái từ $"0"$ lên $"1"$, điều này dẫn đến FF(B) không lật được trạng thái mà vẫn giữ nguyên ở mức $"1"$ và như vậy cũng không làm thay đổi trạng thái đầu ra của FF(C) và FF(D). Q_C vẫn bằng $"0"$.

- Sau xung thứ ba, đầu ra bộ đếm có trạng thái 0011, ($D=0$, $C=0$, $B=1$ và $A=1$).

* Xung đếm C_p thứ tư làm cho Q_A và Q_B chuyển trạng thái từ $"1"$ về $"0"$. Khi Q_B chuyển trạng thái từ $"1"$ về $"0"$ làm cho Q_C chuyển từ $"0"$ lên $"1"$. Điều này có được hoàn toàn giống phân tích ở các trường hợp trên. Mặc dù Q_C chuyển trạng thái nhưng FF(D) vẫn chưa thay đổi trạng thái vì cực tính xung từ Q_C tác động vào FF(D) chưa phù hợp.

Sau xung đếm thứ tư, đầu ra bộ đếm là 0100 ($D=0$, $C=1$, $B = 0$, $A=0$).

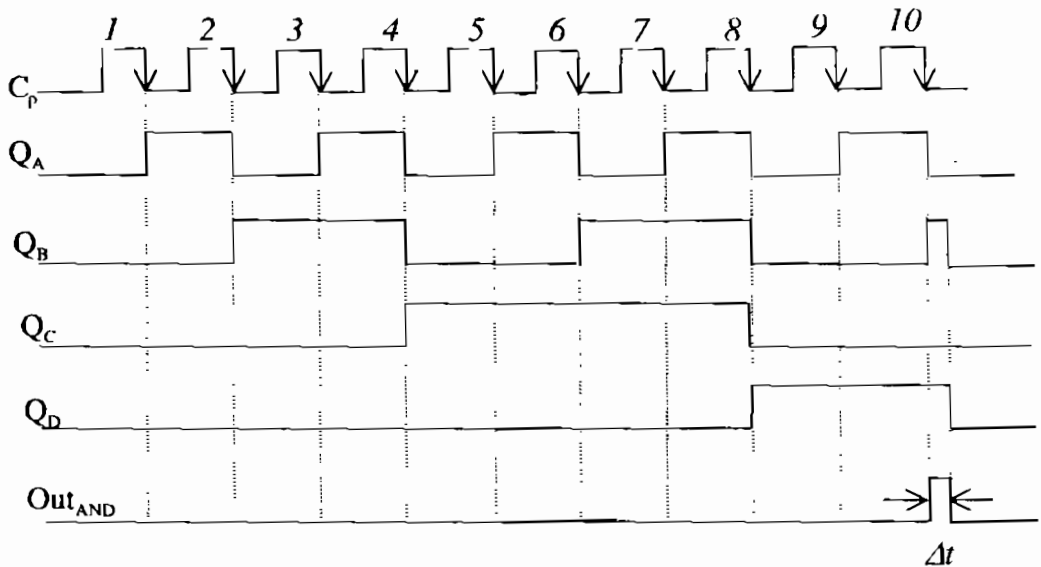
* Quá trình cứ như vậy diễn ra cho tới khi hết xung C_p thứ chín, trạng thái bộ đếm là 1001, ($D=1$, $C=0$, $B=0$ và $A=1$).

* Trong suốt quá trình hoạt động của bộ đếm, từ trạng thái 0000 đến 1001, chúng ta thấy không có xung đếm nào làm cho hai đầu B và D đồng thời = 1 logic. Điều này có nghĩa là đầu ra của cổng AND trong bộ đếm luôn bằng 0 logic, từ trạng thái 0000 đến 1001 của bộ đếm.

* Khi xung đếm thứ mười đưa vào bộ đếm, FF(A) chuyển trạng thái, Q_A từ 1 chuyển về 0, làm Q_B chuyển từ 0 lên 1. Q_B chuyển từ 0 lên 1 không làm thay đổi trạng thái của Q_C và Q_D .

Sau xung đếm thứ 10, đầu ra bộ đếm là 1010. ($D=1$, $C=0$, $B=1$ và $A=0$).

Khi Q_B chuyển trạng thái từ 0 lên 1 và Q_D đã = 1 trước đó, làm đầu ra của cổng AND chuyển trạng thái từ 0 lên 1, tạo ra tín hiệu đưa vào đầu CLR của 4 FF, xoá cả 4 FF về = 0 (trước đó FF(A) và FF(C) đã = 0). Bộ đếm trở về trạng thái 0000 ban đầu sau khi đã đếm đủ 10 xung.



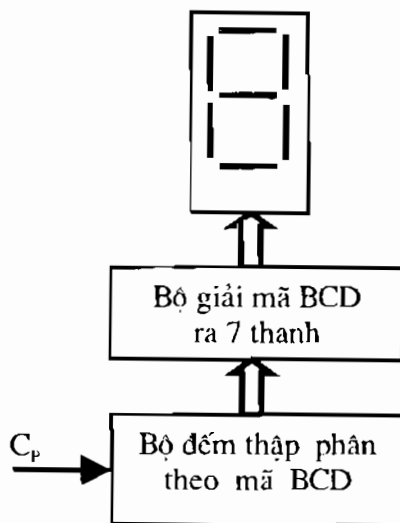
Hình 7.15: Biểu đồ thời gian miêu tả hoạt động của bộ đếm thập phân hoạt động không đồng bộ

Hình 7.15 là biểu đồ mô tả hoạt động của bộ đếm thập phân hoạt động không đồng bộ. Biểu đồ chỉ rõ Q_A lật trạng thái tại thời điểm sườn âm của xung đếm. Q_B lật trạng thái tại thời điểm Q_A từ 1 xuống 0. Q_C lật trạng thái tại thời điểm Q_B từ 1 xuống 0. Q_D lật trạng thái tại thời điểm Q_C từ 1 xuống 0. (Với điều kiện là 2 đầu J và K của các FF đang nối với mức logic 1).

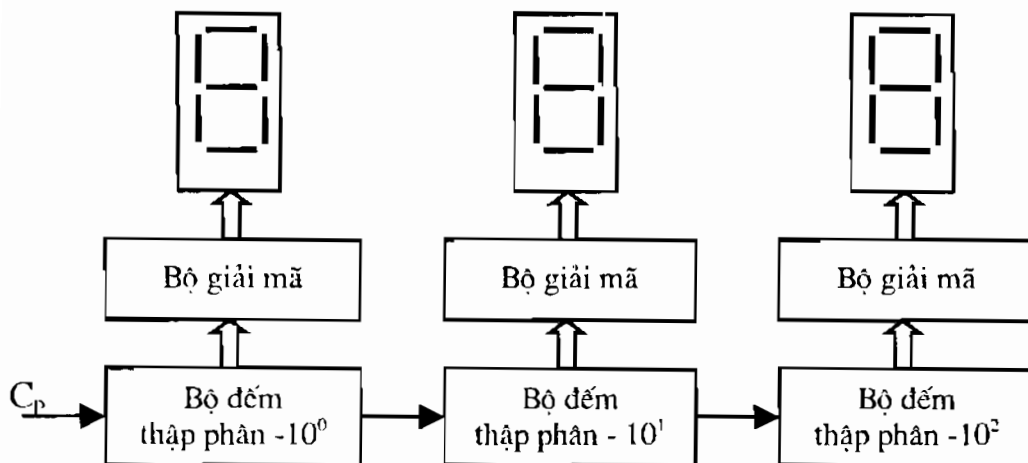
Biểu đồ cũng thể hiện rõ trạng thái đầu ra của cổng AND, nó chỉ ở mức logic 1 khi đồng thời đầu ra Q_B và Q_D ở mức logic 1. Khi đầu ra của cổng AND lên logic 1 nó đặt đầu ra Q của các FF về bằng 0, (bộ đếm về bằng 0).

Δt ở đây chỉ thời gian quá trình quá độ trong mạch.

4. Hiện thị kết quả đếm bằng đèn 7 thanh



Hình 7.16: Sơ đồ khối bộ đếm thập phân hiển thị kết quả đếm bằng đèn 7 thanh.



Hình 7.17: Sơ đồ khối bộ đếm thập phân hiển thị 3 chữ số

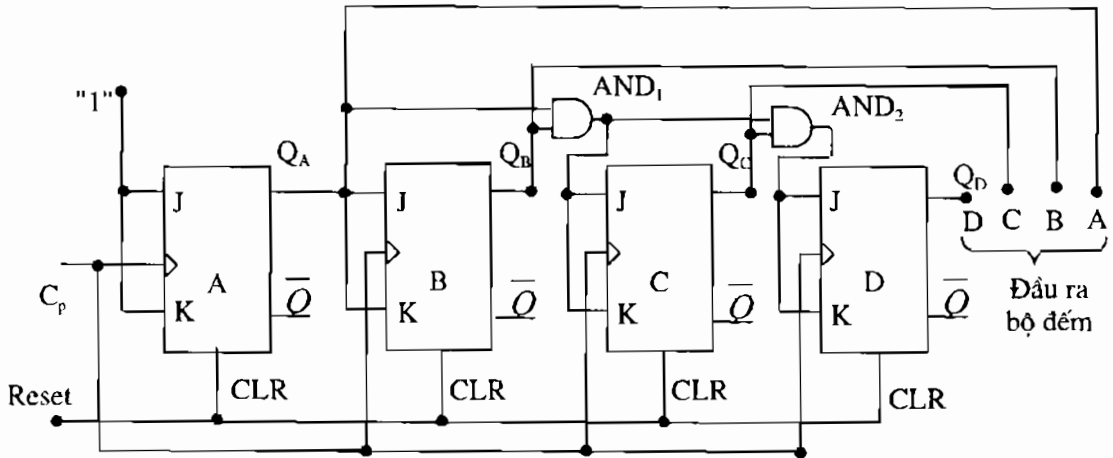
Hình 7.16 vẽ sơ đồ khối mạch đếm thập phân có hiển thị kết quả đếm bằng đèn 7 thanh. Kết quả từ đầu ra của bộ đếm theo kiểu mã BCD - 8421, được đưa vào bộ giải mã BCD ra 7 thanh, từ đầu ra của bộ giải mã tín hiệu được đưa vào kích thích cho đèn hiển thị sáng. Ở sơ đồ khối trên, ứng với mỗi xung đếm đưa vào, đèn hiển thị một chữ số hệ mười. Để hiển thị được nhiều chữ số thập phân hơn cần phải ghép các mạch hình 7.16 như hình 7.17. Ở hình này cho phép

hiển thị được 3 chữ số ở hệ thập phân, các chữ số hệ thập phân có thể hiển thị nằm từ 000 đến 999.

VI. BỘ ĐẾM ĐỒNG BỘ MOD MƯỜI SÁU

Ở phần trên ta đã tìm hiểu về sơ đồ cũng như hoạt động của các bộ đếm hoạt động theo nguyên tắc không đồng bộ. Ở phần này chúng ta tiến hành tìm hiểu về loại bộ đếm hoạt động đồng bộ thông qua việc phân tích sơ đồ và nguyên lý hoạt động của bộ đếm.

1. Sơ đồ logic



Hình 7.18: Sơ đồ bộ đếm Mod 16 hoạt động đồng bộ

Cũng giống như bộ đếm hoạt động không đồng bộ, bộ đếm hoạt động đồng bộ cũng được xây dựng từ các loại FF và các cổng logic.

Hình 7.18 vẽ sơ đồ logic một bộ đếm hoạt động đồng bộ 4 bit, sử dụng JK - FF.

Bộ đếm này có khả năng đếm tối đa là 16 xung. Việc lật trạng thái của các FF diễn ra tại thời điểm sườn đi lên của xung nhịp (nếu điều kiện lật cho phép). Xung đếm được đưa đồng thời vào tất cả các FF trong bộ đếm. Đầu ra của bộ đếm là đầu ra Q của các FF.

Hai đầu J và K của FF(A) - FF có trọng số thấp nhất trong bộ đếm, nối với logic 1. Như vậy, FF này sẽ lật trạng thái mỗi khi có xung đếm đưa vào.

J, K của FF (B) nối trực tiếp với đầu ra của FF(A). Do vậy, khi nào đầu ra $Q_A = \text{logic } 1$ và có sườn dương của xung đếm đưa vào thì FF(B) lật trạng thái.

J, K của FF(C) nối với đầu ra của cổng AND_1 , đầu vào của AND_1 là Q_A và Q_B , như vậy J, K của FF (C) phụ thuộc vào tích $Q_A \cdot Q_B$, khi nào $Q_A \cdot Q_B = "1"$ (làm J, K = "1") và có sườn dương của xung đếm đưa vào thì FF (C) lật trạng thái.

J, K của FF(D) nối với đầu ra của cổng AND_2 . Đầu vào của AND_2 là đầu ra của AND_1 và Q_C , như vậy đầu ra của AND_2 là tích $Q_A \cdot Q_B \cdot Q_C$. Như vậy, ta thấy FF(D) lật trạng thái khi cả 3 FF đứng trước nó phải bằng 1 logic và đồng thời phải có sườn dương của xung đếm.

Qua phân tích trên thể hiện rõ trong bộ đếm đồng bộ, từ FF thứ hai trở đi, việc lật trạng thái của chúng phụ thuộc vào trạng thái đầu ra của các FF đứng trước nó.

2. Nguyên lý hoạt động

- Trước khi đếm, tín hiệu Reset xuất hiện đặt bộ đếm về trạng thái 0 (đầu ra Q của các FF đều bằng 0).

- Xung đếm C_p thứ nhất đưa vào bộ đếm, đặt vào tất cả các FF trong bộ đếm.

- Xung đếm này chỉ làm FF (A) lật trạng thái (Q_A từ 0 đến 1) vì J, K của FF này nối với logic "1".

Các FF khác không lật được trạng thái vì J, K của nó đều đang bằng 0 logic. Sau xung đếm C_p thứ nhất, trạng thái bộ đếm là 0001.

- Xung đếm C_p thứ hai đưa vào bộ đếm FF(A) lại lật trạng thái, Q_A từ 1 $\rightarrow$ 0 do trước đó $Q_A = 1$ nên J, K của FF (B) cũng = 1 logic. Vì vậy, xung C_p thứ hai làm FF(B) lật được trạng thái Q_B từ 0 $\rightarrow$ 1.

Các FF còn lại vẫn không thay đổi trạng thái. Sau xung C_p thứ hai trạng thái bộ đếm là 0010.

- Xung đếm C_p thứ ba đưa vào bộ đếm, FF(A) lật trạng thái, Q_A từ 0 $\rightarrow$ 1.

Do trước khi có xung đếm thứ ba, $Q_A = 0$ nên J, K của FF(B) = 0 vì vậy FF(B) không thay đổi trạng thái, Q_B vẫn bằng 1.

FF(C) có J, K vẫn bằng 0 vì đầu ra cổng $AND_1 = Q_A \cdot Q_B$ mà $Q_A = 0$ nên FF(C) có xung đếm đưa tới vẫn không thay đổi trạng thái. FF(D) tương tự sau xung C_p thứ ba, trạng thái bộ đếm là 0011.

- Xung đếm C_p thứ tư đưa vào bộ đếm Q_A từ 1 $\rightarrow$ 0.

Q_B cũng từ 1 $\rightarrow$ 0 vì trước đó $Q_A = 1$ làm J, K của FF(B) = 1, còn Q_C chuyển từ 0 $\rightarrow$ 1 vì trước xung C_p thứ tư cả Q_A và Q_B đều bằng 1 làm đầu ra của $AND_1 = 1$.

J, K của FF(D) vẫn = 0 vì nó nối với đầu ra của AND_2 , mà đầu ra cổng $AND_2 = Q_A \cdot Q_B \cdot Q_C = 0$ vì Q_C đang = 0. Như vậy, sau xung C_p thứ tư Q_D vẫn = 0. Sau xung thứ tư trạng thái bộ đếm là 0100.

* Quá trình cứ như vậy tiếp diễn.

- Sau 7 xung C_p ta có Q_A, Q_B và Q_C đều = 1. Đầu ra của cổng AND_2 mới = 1 và J, K của FF(D) mới = 1.

- Xung đếm C_p thứ tám lật Q_A, Q_B, Q_C về = 0, Q_D từ 0 lên 1.

Sau xung đếm C_p thứ tám trạng thái bộ đếm là 1000.

Hết xung đếm C_p thứ mười lăm, trạng thái bộ đếm là 1111. Sau xung đếm 16, bộ đếm về 0000.

Hình 7.19 là biểu đồ thời gian mô tả hoạt động của bộ đếm đồng bộ Mod 16.

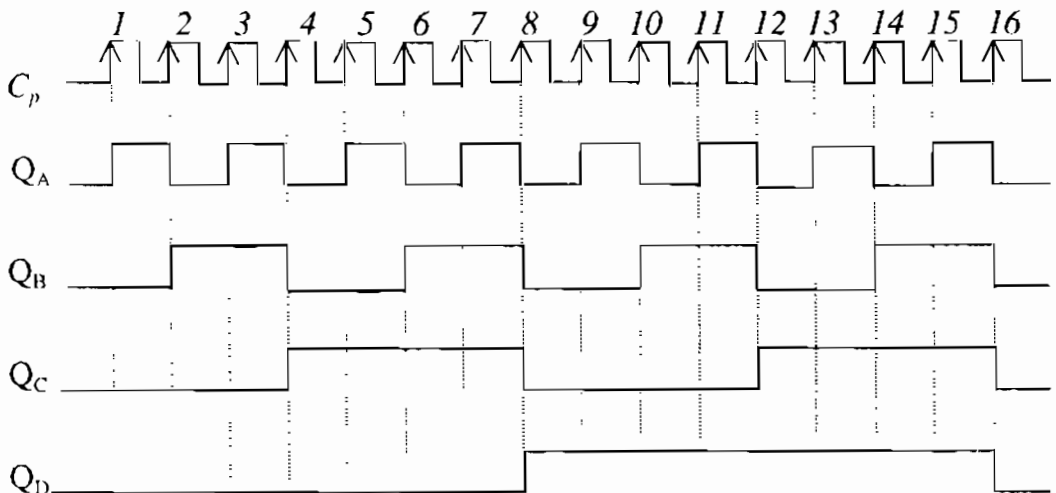
Qua biểu đồ ta thấy Q_A lật trạng thái tại thời điểm sườn lên của xung đếm.

- Q_B lật trạng thái khi có $Q_A = 1$ logic và có sườn lên của xung đếm.

- Q_C lật trạng thái khi có Q_A, Q_B đều = 1 logic và đồng thời có sườn lên của xung đếm.

- Q_D lật trạng thái khi có Q_A, Q_B, Q_C đều = 1 logic và đồng thời có sườn lên của xung đếm.

Sau 16 xung nhịp, bộ đếm trở về trạng thái 0 ban đầu.



Hình 7.19: Biểu đồ thời gian của bộ đếm Mod 16 hoạt động đồng bộ

VII. BỘ ĐẾM THIẾT LẬP TRƯỚC

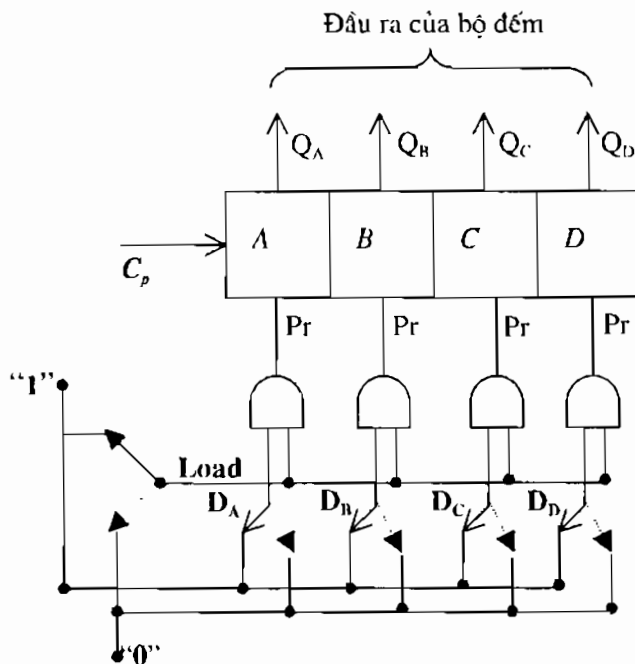
1. Nguyên tắc chung

Một số bộ đếm được thiết kế có khả năng thiết lập trước một trạng thái bất kỳ nào đó, (ngoài trạng thái 0).

Ví dụ hình 7. 20 là bộ đếm tiến 4 bit, có khả năng thiết lập trước trạng thái Bình thường, với bộ đếm 4 bit thì chu trình đếm của bộ đếm này là 16 xung, dung lượng của bộ đếm là 15 xung. Người ta có thể đặt trước trạng thái của bộ đếm để biến đổi nó thành bộ đếm có chu trình là 10 xung, bằng cách đặt cho nó ở trạng thái là $(6)_{10}$ tương ứng ở hệ hai là 0110, thông qua các đầu vào số liệu D_A, D_B, D_C, D_D . Các đầu D_i này được gọi là đầu vào đặt trước. Theo thứ tự D_A có trọng số thấp nhất (2^0), D_D là bit có trọng số cao nhất (2^3). Như vậy, muốn đặt bộ đếm ở $(6)_{10}$ cần đặt $D_A = 0, D_B = 1, D_C = 1$ và $D_D = 0$.

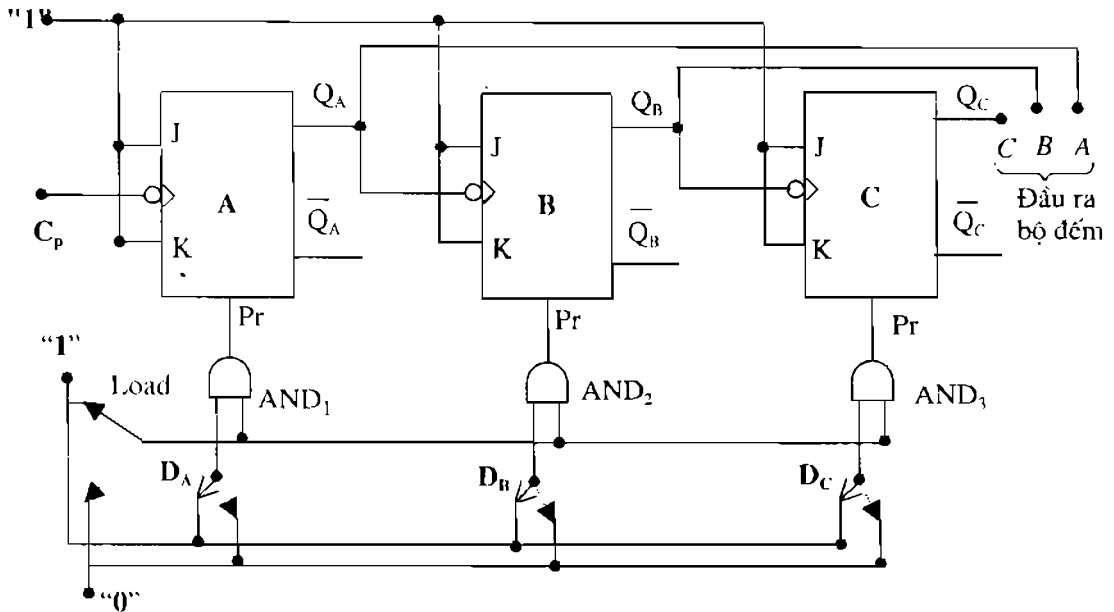
Đầu vào Load có tích cực ở mức cao, nó được dùng để điều khiển nạp số liệu đặt trước vào bộ đếm. Khi Load = 1, các số liệu tương ứng ở D_A, D_B, D_C, D_D sẽ truyền tới các đầu ra tương ứng Q_A, Q_B, Q_C, Q_D .

Sau khi nạp xong phải chuyển Load sang giá trị logic 0 thì bộ đếm mới hoạt động được.



Hình 7.20: Sơ đồ nguyên tắc bộ đếm thiết lập trước

2. Sơ đồ bộ đếm thiết lập trước



Hình 7.21: Sơ đồ bộ đếm thiết lập trước

Hình 7.21 giới thiệu bộ đếm thiết lập trước, 3 bit, hoạt động không đồng bộ, sử dụng JK - FF.

Cũng như bộ đếm 3 bit hoạt động không đồng bộ ta đã xét ở mục *Bộ đếm không đồng bộ Mod 8*, ở đây bộ đếm cũng có khả năng đếm tối đa là 8 xung. (chu trình đếm = 8). Hai đầu J và K của các FF trong bộ đếm cũng được nối với mức logic 1, xung đếm C_p đưa vào FF có trọng số thấp nhất. Việc lật trạng thái của các FF thứ hai trở đi phụ thuộc vào đầu ra của FF đứng trước nó. Đầu ra bộ đếm là đầu ra Q của các FF trong bộ đếm, sau mỗi xung đếm đưa vào, trạng thái bộ đếm lại tăng lên 1.

Điều khác với bộ đếm không đồng bộ Mod 8 là dùng đầu vào hoạt động không đồng bộ, ưu tiên - Pr của các FF để đặt trạng thái cho từng FF tương ứng trước khi đếm. Điều này dẫn tới việc bộ đếm trước khi đếm sẽ ở một trạng thái bất kỳ nào đó, tùy theo yêu cầu sử dụng. Tất nhiên trạng thái đặt trước phải nhỏ hơn trạng thái max của bộ đếm.

Ở hình 7.21 có các cổng AND dùng để dẫn số liệu đặt trước vào từng FF tương ứng. Mỗi cổng AND có hai đầu vào. Đầu vào số liệu D_i và đầu vào tín hiệu điều khiển đặt trước LD $\leftrightarrow$ Load.

3. Phân tích hoạt động

Giả sử ban đầu bộ đếm đang ở trạng thái 000.

D_i được dùng để thiết lập số liệu 0 hoặc 1 thông qua việc đặt các chuyển mạch của D_i tương ứng nối với mức logic 0 hay 1. Số liệu đặt vào bộ đếm là bao nhiêu phụ thuộc vào yêu cầu sử dụng bộ đếm theo mod nào.

Sau khi đã đặt các chuyển mạch D_i ở các vị trí tương ứng với số liệu cần đưa vào, tiến hành đặt cho tín hiệu Load ở mức logic 1. Như vậy các số liệu vào sẽ đi qua cổng AND đưa tới đầu Pr của các FF, đầu ra Q tương ứng có trạng thái theo đúng số liệu đặt vào. Khi số liệu đã đưa tới đầu ra Q của các FF cần phải đưa tín hiệu Load về mức logic 0, có như vậy Pr của các FF lúc đó mới không có tác dụng, bộ đếm mới hoạt động được (vì theo sơ đồ hình 7. 21 Pr có mức tích cực dương, tương đương 1 logic).

Để hiểu rõ hơn, chúng ta tiến hành phân tích một trường hợp cụ thể về hoạt động của bộ đếm thiết lập trước trạng thái đếm, dựa theo sơ đồ hình 7. 21.

Giả sử ta muốn có bộ đếm, đếm được 5 xung. Chu trình đếm tối đa của bộ đếm 3 bit là 8 xung. Vậy trạng thái cần đặt trước là $8 - 5 = 3$. Tương ứng phải đặt chuyển mạch D_A và chuyển mạch D_B nối với logic 1 còn chuyển mạch D_C nối với logic 0 (011) $\leftrightarrow (3)_{10}$. (Giả sử trước khi thiết lập, trạng thái bộ đếm đang là 000).

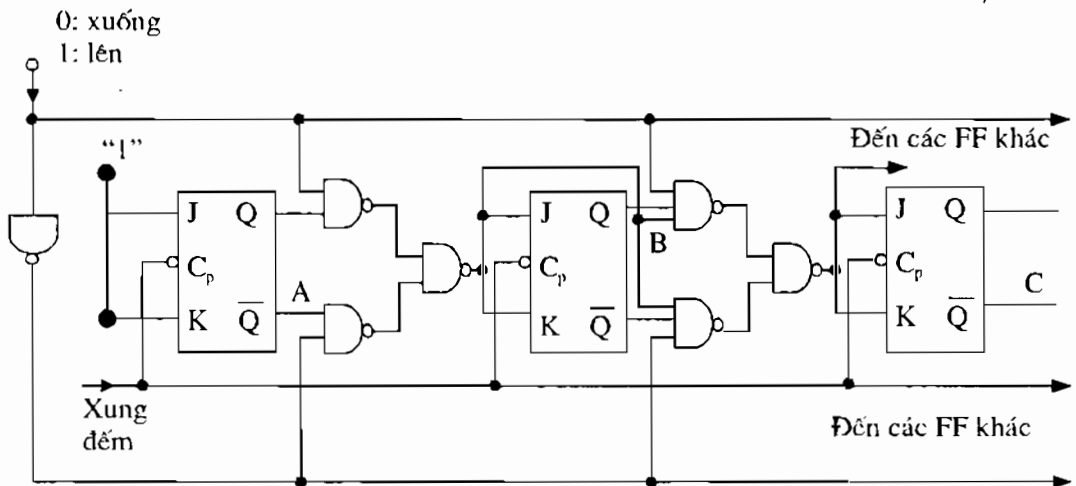
Tiếp theo đặt tín hiệu Load lên 1 logic, khi đó đầu ra của mỗi cổng AND có giá trị tương ứng với các giá trị của D_i đưa vào mỗi cổng, (vì cổng AND có 2 đầu vào, một đầu vào = 1 logic, đầu ra của AND sẽ là giá trị của đầu vào còn lại). Giá trị từ đầu ra của cổng AND đưa vào Pr đặt Q theo trạng thái điều khiển của Pr. Do đó, đầu ra Q_A , Q_B chuyển từ 0 logic lên 1 logic, Q_C vẫn giữ nguyên 0 logic, đầu ra bộ đếm sẽ có trạng thái 011 ($C = 0, B = 1, A = 1$).

Sau khi đầu ra Q của các FF đã ở mức logic tương ứng với số liệu đặt vào, chuyển tín hiệu Load từ 1 logic về 0 logic, 3 đầu ra của 3 cổng AND đều bằng 0, đầu Pr của cả 3 FF bằng 0 logic. Bộ đếm bắt đầu hoạt động đếm theo xung đếm C_p đưa vào.

Như vậy, bộ đếm xuất phát đếm không phải từ giá trị 000 mà từ 011, vì vậy chỉ cần 5 xung đếm nữa là bộ đếm về trạng thái ban đầu, (000). Việc hoạt động đếm diễn ra trong mạch đếm hoàn toàn giống như ở bộ đếm không đồng bộ Mod 8 ta đã tìm hiểu.

Câu hỏi và bài tập

1. Nêu định nghĩa, phân loại và công dụng của bộ đếm.
2. Phân tích đặc điểm sơ đồ và nguyên lý hoạt động của bộ đếm tiến, hoạt động không đồng bộ Mod tám dùng JK - FF.
3. Phân tích đặc điểm sơ đồ và nguyên lý hoạt động của bộ đếm lùi, hoạt động không đồng bộ Mod tám JK - FF.
4. Phân tích đặc điểm sơ đồ và nguyên lý hoạt động của bộ đếm tiến, lùi hoạt động không đồng bộ Mod tám dùng JK - FF.
5. Phân tích đặc điểm sơ đồ và nguyên lý hoạt động của bộ đếm tiến hoạt động không đồng bộ Mod mười, loại dùng JK - FF và loại dùng T- FF.
6. Phân tích đặc điểm sơ đồ và nguyên lý hoạt động của bộ đếm tiến hoạt động đồng bộ Mod mười sáu.
7. Phân tích sơ đồ của bộ đếm tiến Mod mười hoạt động đồng bộ, dùng JK - FF.
8. Phân tích sơ đồ và nguyên lý hoạt động của bộ đếm thiết lập trước.
9. Phân tích hoạt động của mạch theo sơ đồ hình 7.22. Chỉ ra chức năng và đặc điểm của mạch.



Hình 7. 22

Chương 8

BỘ GHI DỊCH

Mục tiêu

- Phân tích được đặc điểm sơ đồ và nguyên lý làm việc của các bộ ghi dịch nạp thông tin nối tiếp và song song.
- Vận dụng kiến thức để phân tích được các mạch điện trong các môn học tiếp theo.
- Hình thành được tính sáng tạo, tư duy logic và khoa học.

Nội dung trọng tâm

- Khái niệm về bộ ghi dịch.
- Bộ ghi dịch, nạp thông tin vào nối tiếp.
- Bộ ghi dịch, nạp thông tin vào song song.

I. KHÁI NIỆM

1. Định nghĩa

Bộ ghi dịch (còn gọi là thanh ghi dịch) có khả năng tiếp nhận, lưu trữ và dịch chuyển thông tin (dịch phải hoặc dịch trái).

2. Phân loại

Có nhiều cách phân loại bộ ghi dịch.

2.1. Phân loại theo cách đưa thông tin vào và lấy thông tin ra

- Vào nối tiếp, ra song song: Thông tin được đưa vào thanh ghi tuần tự từng bit một, số liệu được đưa ra đồng thời, tức là tất cả n FF của thanh ghi được đọc cùng một lúc.
- Vào song song, ra song song: Thông tin được đưa vào và lấy ra đồng thời ở n FF.
- Vào nối tiếp, ra nối tiếp: Thông tin được đưa vào, lấy ra tuần tự từng bit một.

- Vào song song, ra nối tiếp: Thông tin được đưa vào đồng thời cả n FF, lấy ra tuần tự từng bit một dưới sự điều khiển của xung nhịp.

2.2. Phân loại theo đầu vào

- Đầu vào đơn: Mỗi FF trong bộ ghi dịch chỉ sử dụng một đầu vào điều khiển, ví dụ đầu vào điều khiển D của D-FF hoặc các loại FF khác mắc theo kiểu D.

- Đầu vào đôi: Các FF trong bộ ghi dịch sử dụng cả hai đầu vào điều khiển, ví dụ hai đầu vào điều khiển J, K hay R, S của 2 loại FF.

2.3. Phân loại theo đầu ra

- Đầu ra đơn: Mỗi FF trong bộ ghi dịch chỉ có một đầu ra Q_i (hoặc $\overline{Q}_i$) được đưa ra chân của vi mạch.

- Đầu ra đôi: Cả hai đầu ra của FF là Q_i và $\overline{Q}_i$ đều được đưa ra chân của vi mạch.

3. Công dụng

Bộ ghi dịch được sử dụng rộng rãi để nhớ số liệu, chuyển số liệu song song thành nối tiếp, nối tiếp thành song song. Bộ ghi dịch là thành phần không thể thiếu được trong các bộ vi xử lý, trong các cổng vào ra có khả năng lập trình.

Bộ ghi dịch còn được dùng để thiết kế bộ đếm, tạo dãy tín hiệu nhị phân tuần hoàn theo yêu cầu cho trước.

II. BỘ GHI DỊCH NẠP THÔNG TIN VÀO NỐI TIẾP

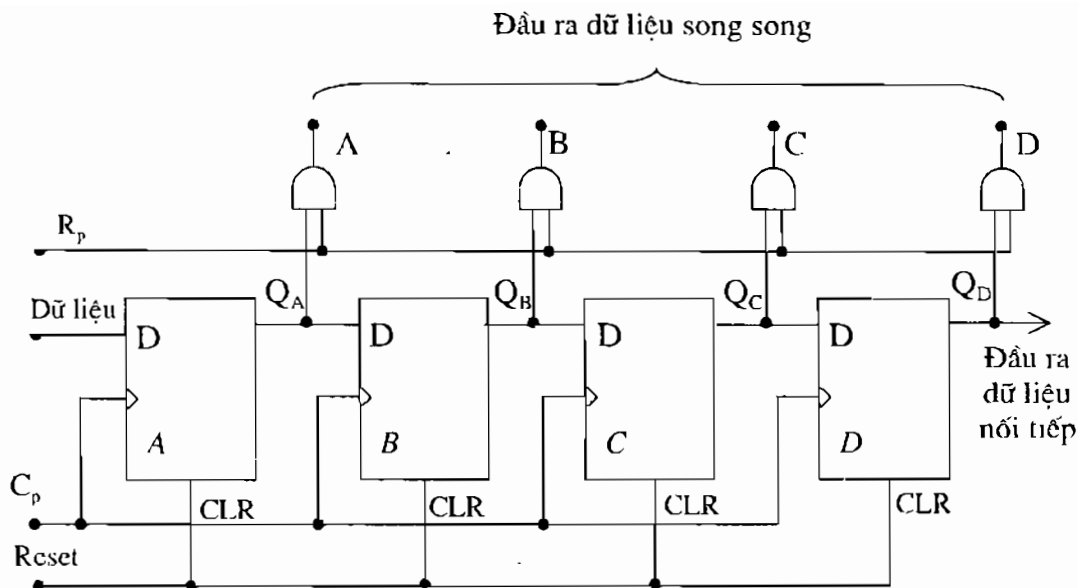
1. Định nghĩa

Là mạch điện tử số có khả năng tiếp nhận thông tin vào nối tiếp từng bit một, lưu trữ và dịch chuyển được thông tin (có thể dịch phải, dịch trái hoặc dịch theo cả hai hướng). Bộ ghi dịch được xây dựng từ các FF và các cổng logic.

2. Sơ đồ bộ ghi dịch

Hình 8.1 giới thiệu sơ đồ bộ ghi dịch nạp, thông tin vào nối tiếp 4 bit.

Bộ ghi dịch này lưu trữ được 4 bit dữ liệu. Nó được xây dựng từ các D - FF loại hoạt động đồng bộ.



Hình 8.1: Sơ đồ bộ ghi dịch nối tiếp

D-FF(A) là bit có trọng số thấp nhất (2^0), D - FF(D) là bit có trọng số cao nhất (2^3). Dữ liệu vào đưa vào qua đầu D của FF có trọng số thấp nhất (FF(A)) bằng các xung nhịp C_p .

Từ đầu ra Q của FF có trọng số thấp được nối vào đầu D của FF có trọng số cao hơn liền kề.

C_p dùng để nạp dữ liệu vào và lấy dữ liệu ra theo kiểu nối tiếp. Với thanh ghi 4 bit cần 4 xung nhịp C_p để nạp dữ liệu vào và cũng cần 4 xung nhịp C_p để lấy dữ liệu ra theo kiểu nối tiếp. Thanh ghi có n bit cần n xung C_p để nạp và để đọc ra nối tiếp. Dữ liệu lấy ra nối tiếp ở đầu ra của FF có trọng số lớn nhất. Ngoài ra, có thể lấy dữ liệu ra theo kiểu song song bằng cách đưa tín hiệu $R_p = 1$ vào bộ ghi dịch. Khi không đọc dữ liệu ra theo kiểu song song R_p đặt ở "0".

Một đặc điểm của bộ ghi dịch ở hình 8.1 là khi lấy dữ liệu ra theo kiểu nối tiếp thì sau khi lấy xong dữ liệu, trong thanh ghi dữ liệu cũng bị xoá hết. Còn lấy ra song song dữ liệu không bị mất đi.

Tín hiệu Rreset dùng để xoá thanh ghi về trạng thái 0.

3. Nguyên lý hoạt động

Để tìm hiểu về hoạt động nạp dữ liệu và cũng như lấy dữ liệu ra khỏi bộ ghi dịch, ta phân tích một ví dụ sau.

Ví dụ: Cần nạp dữ liệu 1011 vào thanh ghi. Dữ liệu được đưa vào đầu vào dữ liệu của bộ ghi dịch.

Trước khi nạp, tín hiệu Reset đặt thanh về bằng 0. (Thực tế nếu thanh ghi cấu tạo như hình 8.1 có thể không cần tín hiệu Reset).

- Xung nhịp C_p thứ nhất đưa vào các FF của bộ ghi dịch, nó nạp bit có trọng số lớn nhất của dữ liệu vào FF(A). Q_A từ 0 $\rightarrow$ 1. FF(B) tiếp nhận giá trị của Q_A trước đó vì vậy $Q_B = 0$, tương ứng Q_C, Q_D cũng bằng 0.

Sau xung nhịp thứ nhất, trong thanh ghi có nội dung 0001.

- Xung nhịp C_p thứ hai đưa vào bộ ghi dịch, xung nhịp này có tác dụng nạp bit dữ liệu thứ hai sau bit có trọng số lớn nhất vào FF(A), $Q_A = 0$. Giá trị trước đó của Q_A được chuyển sang FF(B) nên $Q_B = 1$, giá trị trước đó của FF(B) chuyển sang FF(C) và của FF(C) chuyển sang FF(D) của FF(D) bị mất đi.

Sau xung C_p thứ hai trong bộ ghi có giá trị là 0010.

- Xung nhịp C_p thứ ba nạp bit dữ liệu thứ ba vào bộ ghi tương tự như 2 xung C_p trên, sau xung C_p thứ ba nội dung trong bộ ghi là 0101.

- Xung nhịp C_p thứ tư nạp bit dữ liệu cuối cùng (bit có trọng số nhỏ nhất) vào bộ ghi dịch.

Sau xung nhịp này, trong bộ ghi có giá trị 1011.

Như vậy, sau 4 xung nhịp C_p , dữ liệu đã được nạp vào bộ ghi dịch.

Bộ ghi dịch này nạp dữ liệu từ bit có trọng số cao nhất đến bit có trọng số thấp nhất. (Dịch trái dữ liệu).

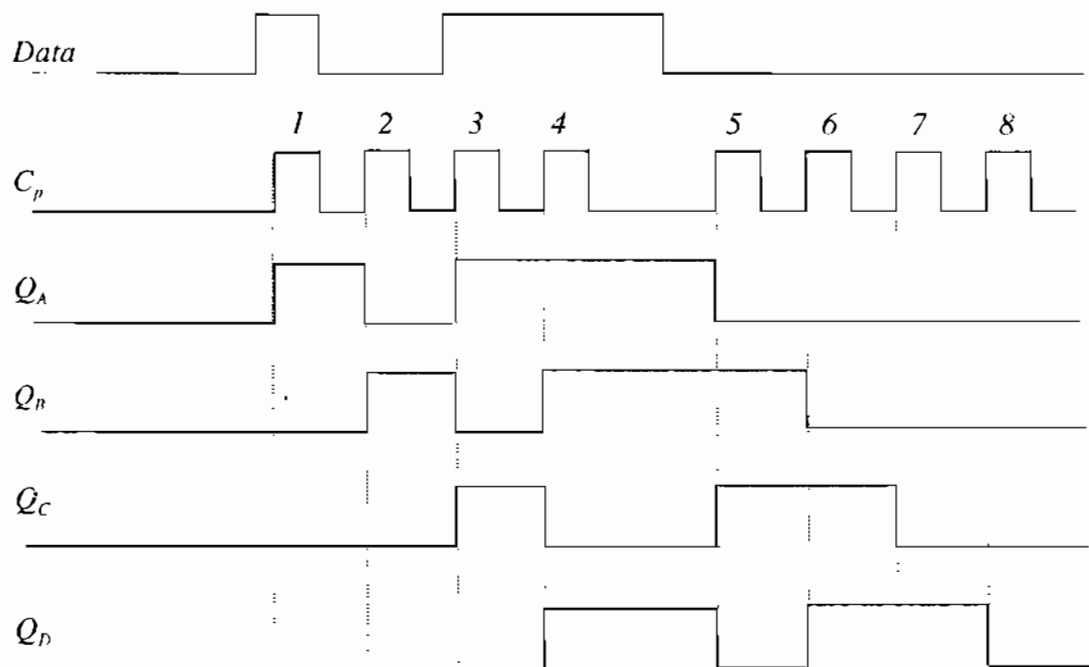
- Để đọc dữ liệu ra song song, đặt R_p lên mức logic 1. Tín hiệu này đặt vào các cổng AND1 $\rightarrow$ AND4. Đầu ra của cổng AND chính là giá trị có ở đầu vào còn lại của nó (cổng AND có 2 đầu vào, một đầu ở mức logic 1, đầu ra là trạng thái của đầu vào còn lại). Đầu vào thứ hai của các cổng AND lấy tín hiệu từ các đầu ra tương ứng của các FF trong thanh ghi. Đó chính là các dữ liệu có trong thanh ghi. Như đã nói ở trên, với cách đọc dữ liệu ra theo kiểu song song này, dữ liệu trong thanh ghi sẽ không bị mất đi (với điều kiện nguồn cung cấp cho thanh ghi làm việc không bị mất). Sau khi đọc xong dữ liệu ra, phải đặt $R_p = 0$.

- Đọc dữ liệu từ thanh ghi ra theo kiểu nối tiếp: -

Dùng xung nhịp C_p đưa vào thanh ghi. Mỗi xung nhịp đưa vào sẽ đẩy thông tin dịch trái một bit. Dữ liệu được lấy ra từ bit có trọng số lớn nhất. Sau 4 xung

nhịp, dữ liệu có trong thanh ghi đã được đọc ra đầy đủ. Nội dung trong thanh ghi cũng bị mất đi.

Hình 8.2 vẽ biểu đồ mô tả việc nạp dữ liệu 1011 vào thanh ghi 4 bit, và lấy dữ liệu ra theo kiểu nối tiếp.



Hình 8.2: Biểu đồ thời gian của bộ ghi dịch nối tiếp

III. BỘ GHI DỊCH NẠP THÔNG TIN VÀO SONG SONG

1. Nhiệm vụ

Tiếp nhận dữ liệu vào theo kiểu song song, lưu trữ và dịch chuyển dữ liệu. Đưa dữ liệu ra theo kiểu nối tiếp và song song.

2. Sơ đồ

Hình 8.3 giới thiệu sơ đồ bộ ghi dịch nạp dữ liệu vào song song 4 bit.

Mạch được xây dựng từ loại JK-FF. Có thể dùng loại FF khác cũng xây dựng được bộ ghi dịch với nhiệm vụ như trên. Trong sơ đồ này, FF(A) là bit có trọng số nhỏ nhất, FF(D) là bit có trọng số lớn nhất.

- Đầu ra Q của các FF có trọng số thấp nối vào đầu J của FF có trọng số cao hơn liền kề.

- Đầu ra $\overline{Q}$ nối với đầu K của FF liền kề giúp cho dữ liệu được dịch từ bit có trọng số thấp sang bit có trọng số cao hơn sau mỗi xung nhịp.

- Đầu ra Q của FF có trọng số lớn nhất nối vào đầu vào J của FF có trọng số nhỏ nhất. Đầu ra $\overline{Q}$ của FF có trọng số cao nhất nối vào đầu vào K của FF có trọng số nhỏ nhất. Việc nối như vậy sẽ làm dữ liệu trong thanh ghi không bị mất đi khi dịch dữ liệu sang trái.

Dữ liệu nạp vào song song được đưa vào đầu vào không đồng bộ ưu tiên P_r của các FF trong bộ ghi.

Đầu ra dữ liệu song song là đầu Q của các FF cho qua các cổng AND1 đến AND4. Lấy dữ liệu ra song song bằng cách đặt $R_p = 1$.

Lấy dữ liệu ra nối tiếp ở đầu Q của FF(D) là FF có trọng số lớn nhất bằng cách dùng xung nhịp đọc ra từng bit.

Tín hiệu Reset đặt bộ ghi dịch về = 0.

3. Nguyên lý hoạt động

** Nạp dữ liệu vào bộ ghi dịch:*

- Tín hiệu Reset đặt bộ ghi về = 0.

- Dữ liệu đưa đồng thời vào các đầu D $\rightarrow$ A tới đầu P_r của các FF tương ứng và lật trạng thái theo đúng giá trị của dữ liệu nạp vào.

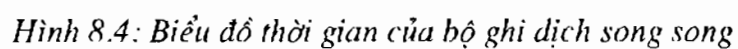
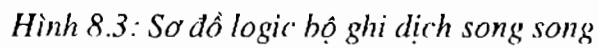
Giả sử đặt dữ liệu cần nạp 1010 vào D $\rightarrow$ A, FF(D) và FF(B) lật trạng thái từ 0 $\rightarrow$ 1 còn FF(C) và FF(A) không thay đổi trạng thái. Sau khi nạp xong dữ liệu, trong bộ ghi có $Q_D = 1$, $Q_C = 0$, $Q_B = 1$, $Q_A = 0$.

** Đọc dữ liệu ra theo kiểu song song:*

Tương tự như có bộ ghi dịch nối tiếp. Để đọc dữ liệu từ bộ ghi dịch ra theo kiểu song song cần đặt tín hiệu R_p ở mức "1". Với mức tín hiệu này, dữ liệu từ đầu ra Q của các FF trong bộ ghi dịch sẽ được đưa qua cổng AND và đưa ra ngoài.

** Đọc dữ liệu ra theo kiểu nối tiếp:*

Tương tự cách đọc nối tiếp ở bộ ghi dịch nạp nối tiếp. Điều khác ở đây là do có vòng hồi tiếp từ Q_D nối vào J_A và $\overline{Q_D}$ nối K_A nên khi lấy dữ liệu ra, nội dung dữ liệu trong thanh ghi không bị mất đi.



Hình 8.4 vẽ biểu đồ thời gian mô tả việc nạp dữ liệu vào song song và lấy dữ liệu ra nối tiếp.

Dữ liệu vào đặt tới đầu Pr của các FF qua đầu vào dữ liệu song song A, B, C, D. ($A=1, B=0, C=1, D=1$). Với dữ liệu vào như vậy làm đầu ra Q của các FF trong mạch thay đổi trạng thái. (Giả sử trạng thái Q của các FF trước khi có dữ liệu đều đang = 0). Sau khi có dữ liệu vào, $Q_A=1, Q_B=0, Q_C=1$ và $Q_D=1$.

Khi có xung nhịp C_p đưa vào, mỗi xung C_p làm dữ liệu trong FF dịch sang trái 1 bit. Sau 4 xung C_p trạng thái của FF lại trở lại như trước khi có C_p thứ nhất.

Câu hỏi và bài tập

1. Trình bày định nghĩa, các cách phân loại và công dụng của bộ ghi dịch.
2. Phân tích đặc điểm sơ đồ và nguyên lý làm việc của bộ ghi dịch nạp thông tin vào nối tiếp dùng D - FF.
3. Vẽ sơ đồ logic, phân tích sơ đồ và hoạt động của bộ ghi dịch nạp thông tin vào nối tiếp dùng RS - FF.
4. Phân tích đặc điểm sơ đồ và nguyên lý làm việc của bộ ghi dịch nạp thông tin vào song song và nối tiếp dùng JK - FF.
5. Vẽ sơ đồ logic, phân tích sơ đồ và hoạt động của bộ ghi dịch nạp thông tin vào song song dùng T - FF.
6. Dùng T - FF vẽ sơ đồ bộ ghi dịch có khả năng nạp thông tin vào nối tiếp và vào song song. Giải thích hoạt động của sơ đồ.

Chương 9

BỘ NHỚ

Mục tiêu

- Nắm được các khái niệm về bộ nhớ.
- Phân tích được cấu trúc, hoạt động và các đặc tính cơ bản của các loại bộ nhớ bán dẫn (ROM và RAM).
- Hình thành được tích cách tư duy logic và yêu thích nghề nghiệp.

Nội dung trọng tâm

- Khái niệm chung về bộ nhớ.
- Cấu trúc và hoạt động của các loại bộ nhớ bán dẫn.
- Các đặc tính cơ bản của bộ nhớ bán dẫn.

I. KHÁI NIỆM VỀ BỘ NHỚ

Đối với các thiết bị số, khả năng lưu trữ được dữ liệu là một việc rất cần thiết. Chẳng hạn trong máy tính, các con số cần thiết cho phép toán phải được lưu trữ ngay trong máy. Còn trong các thiết bị điều khiển số thì lệnh điều khiển cũng phải được lưu trữ để thực hiện dần theo một trình tự nào đó. Vì vậy, bộ nhớ là một thành phần không thể thiếu được của các thiết bị số.

Khi nghiên cứu về bộ nhớ, ta lưu ý rằng, thông tin hay dữ liệu lưu chuyển trong các thiết bị số đều phải ở dưới dạng mã hệ 2, tức là một chuỗi kết hợp bởi 0 và 1, và được biểu diễn bởi hai mức điện thế khác nhau.

1. Từ nhớ (word)

Thông thường, thông tin hay dữ liệu được tạo thành từ một đơn vị cơ bản gọi là từ (word). Một từ có chiều dài nhất định tùy theo loại máy, chẳng hạn là 8 bit, 12 bit, 16 bit, 32 bit, v.v... Từ là thành phần thông tin cơ bản nhất. Các bộ phận của thiết bị thường chỉ truyền đi hay nhận vào nguyên một từ (hay nhiều từ) chứ không phải vài bit của từ. Tuy nhiên, vì từ được tạo thành từ nhiều bit nên đơn vị cơ bản của bộ nhớ chính là đơn vị nhớ lưu trữ được một bit.

Khi so sánh các bộ nhớ, người ta thường lưu ý đến những đặc tính sau đây:

2. Dung lượng (capacity)

Dung lượng (hay nói rõ hơn là dung lượng nhớ) là khối lượng thông tin hay dữ liệu có thể lưu trữ được trong bộ nhớ. Để xác định dung lượng, ta thường dùng đơn vị là bit (hoặc Kilobit hoặc Megabit). Dung lượng liên quan mật thiết đến giá thành của bộ nhớ. Giá thành này thường được đánh giá theo tiêu chuẩn: chi phí/bit.

3. Thời gian thâm nhập (access time)

Thời gian này gồm có hai phần: thứ nhất là phần thời gian cần thiết để xác định vị trí của từ (thời gian tìm từ) trong bộ nhớ và thứ hai là phần thời gian cần thiết để lấy từ ra khỏi bộ nhớ. Thời gian thâm nhập (viết tắt là at) là một thông số quan trọng của bộ nhớ, nếu nó kéo dài thì sẽ làm giảm khả năng làm việc của thiết bị, vì thiết bị chỉ hoạt động được chừng nào chúng nhận được dữ liệu mà thôi.

II. BỘ NHỚ RAM

Thuật ngữ RAM là viết tắt của từ tiếng Anh: Random Access Memory, thường được dùng để chỉ các bộ nhớ đọc - viết.

Bộ nhớ RAM thường được sử dụng trong các thiết bị tính để cất giữ các kết quả trung gian hay kết quả tạm thời trong khi thực hiện các chương trình điều khiển.

Như ta đã biết, một mạch FF có hai trạng thái, rất tiện dụng để làm một đơn vị nhớ của bộ nhớ đọc viết. Tuy nhiên, trong quá khứ, ý tưởng này không được thực hiện vì mạch FF làm bằng bộ phận rời có kích thước tương đối lớn, tiêu thụ công suất đáng kể và tốn kém. Nhưng từ khi kỹ thuật mạch tích hợp ra đời, người ta đã chế tạo ra được các mạch tích hợp với nhiều bộ phận bán dẫn trên một diện tích silic ngày càng bé. Từ đó bộ nhớ bán dẫn dùng FF ra đời.

Hiện nay, có hai loại bộ nhớ bán dẫn RAM chính là loại dùng transistor lưỡng cực và loại MOSFET.

- + Bộ nhớ RAM dùng transistor lưỡng cực lấy FF làm đơn vị nhớ cơ bản vận tốc rất cao, at vào khoảng vài chục nanosec và công suất tiêu thụ vào khoảng 1 nanowatt/bit.

- + Bộ nhớ RAM dùng MOSFET được chia làm hai loại:

- Loại tĩnh (static) cũng lấy cấu trúc FF làm đơn vị nhớ cơ bản.

- Loại động (dynamic) lợi dụng điện dung ký sinh của cực cổng (gate) để chứa dữ liệu.

Sở dĩ người ta gọi tĩnh và động là vì loại dùng mạch FF không cần xung kích thích để tăng cường thêm điện tích trong tụ điện (thao tác "làm tươi bộ nhớ"), nếu không điện tích này sẽ giảm đi, làm mất dữ liệu. Nói chung, bộ nhớ MOSFET chậm hơn bộ nhớ lưỡng cực nhưng bù lại, đơn vị nhớ kích thước nhỏ hơn nên có thể thu gọn nhiều đơn vị nhớ trong một diện tích silic nhỏ mà công suất tiêu thụ lại không đáng kể (chẳng hạn 250mW cho 4096 bit đối với các bộ nhớ thế hệ đầu tiên).

Khi cần các bộ nhớ RAM có dung lượng lớn, ta có thể mắc nhiều bộ nhớ nhỏ lại với nhau mà kích thước toàn thể không lớn lắm.

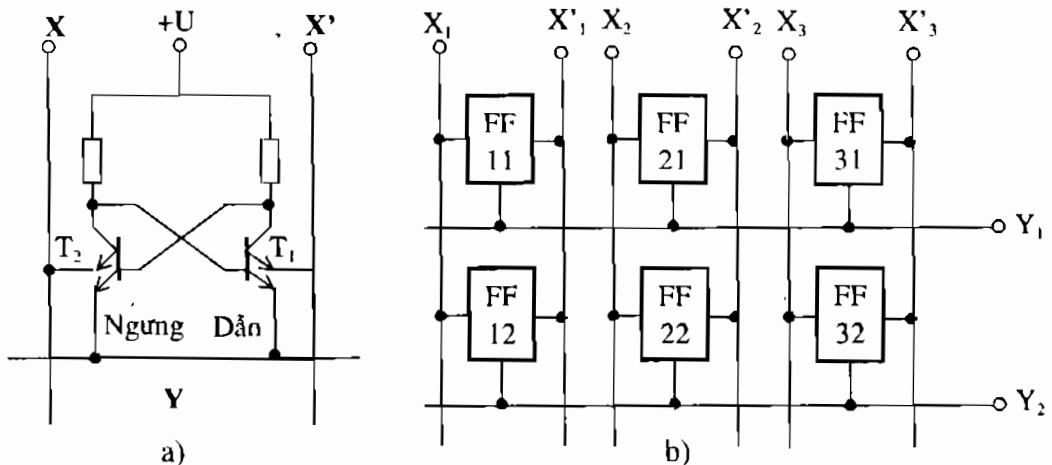
Sau đây, ta sẽ khảo sát chi tiết hơn về các bộ nhớ bán dẫn RAM.

1. Bộ nhớ RAM dùng transistor lưỡng cực

1.1. Đơn vị nhớ

Đơn vị nhớ (memory cell) là một FF nhưng được biến đổi đi đôi chút như sơ đồ trên hình 9.1.a: Transistor có nhiều cực phát (như TTL), mạch có hai đường XX' và một đường Y.

Mỗi đường Y dùng cho một từ, còn cặp XX' dùng cho từng bit của từ. Như vậy, muốn chọn từ nào và bit nào ta sẽ kích thích các đường Y, XX' tương ứng. Trên hình 9.1.b là một mạch nhớ có 6 đơn vị nhớ được tạo thành từ 6 FF. Đường Y₁ dùng cho từ thứ nhất (có 3 đơn vị nhớ FF 11, FF 21 và FF 31), còn đường Y₂ dùng cho từ thứ hai (có 3 đơn vị nhớ FF 12, FF 22 và FF 32).



Hình 9.1: Đơn vị nhớ RAM dùng transistor lưỡng cực

Bình thường thì đường Y ở điện thế thấp (chẳng hạn 0V) còn hai đường XX' ở điện thế cao (+U). Do đó, hai cực phát nối vào dây XX' không dẫn điện, FF hoạt động nhờ hai cực phát nối vào dây Y và duy trì một trạng thái bền nào đó, chẳng hạn T_1 dẫn, T_2 ngưng.

+ *Thao tác đọc*

Muốn đọc một từ ta nâng điện thế của đường Y tương ứng lên cao hơn $+U_x$ làm cho hai cực phát nối vào đường này ngưng dẫn và FF phải hoạt động nhờ vào hai cực phát nối vào XX'. Lúc này dòng điện sẽ chảy qua transistor T_1 (dẫn) để vào X' và ra ngoài. Sự hiện diện của dòng điện báo hiệu sự xuất hiện của tín hiệu:

- Nếu có dòng điện chảy trong đường X' (T_1 dẫn) thì ta bảo FF chứa bit 0.
- Nếu có dòng điện chảy trong đường X (T_2 dẫn) thì ta bảo FF chứa bit 1.

Mạch ngoài sẽ có nhiệm vụ biến dòng điện thành điện thế thích hợp cho các cổng logic (thường là TTL). Sau khi đọc xong, đường Y lại trở về điện thế thấp, như vậy ta chỉ đưa vào Y một xung điện thế để đọc và nhận được một xung dòng điện chạy ra. Phép đọc như vậy không làm mất dữ liệu nên không cần phải viết lại.

+ *Thao tác viết*:

Muốn viết một từ ta phải qua hai giai đoạn: Trước hết ta phải nâng điện thế của đường Y tương ứng để lựa chọn từ. Sau đó điều khiển các đường XX' để viết các bit vào như sau:

- Muốn viết bit 0 ta hạ điện thế đường X' thấp hơn đường X trong một thời gian ngắn làm T_1 dẫn.
- Muốn viết bit 1 ta làm ngược lại, nghĩa là hạ điện thế X thấp hơn đường X' làm cho T_2 dẫn.

Điều nói trên có nghĩa là ta sẽ cho vào X hay X' một xung hướng âm.

1.2. Đặc tính

Bộ nhớ RAM dùng transistor lưỡng cực có các đặc tính sau:

- Vận tốc cao, thời gian thâm nhập bé (khoảng chừng vài chục nanosec) nên thuận tiện cho việc sử dụng trực tiếp với các bộ phận xử lý dữ liệu.
- Vì luôn có transistor ở trạng thái dẫn nên luôn tiêu thụ công suất. So với loại MOSFET thì công suất tiêu thụ trên 1 bit luôn luôn lớn hơn.

- Những mạch có dung lượng lớn thường có thêm mạch giải mã để tìm địa chỉ của từ và các mạch đầu ra để làm tăng hệ số mắc tải (fan out), tất cả được đặt trên cùng một miếng bán dẫn.

- Có một vài biến thể của đơn vị nhớ dùng transistor lưỡng cực. Chẳng hạn như loại dùng diode Schottky. Loại này có một số ưu điểm so với đơn vị nhớ dùng transistor lưỡng cực phát như công suất tiêu tán nhỏ hơn (khi có cùng điện trở cực góp R_c). Ngoài ra, có thể giảm nhỏ công suất tiêu tán bằng cách tăng R_c , tuy nhiên trong trường hợp như vậy, thời gian thâm nhập sẽ tăng lên.

2. Bộ nhớ RAM dùng MOSFET

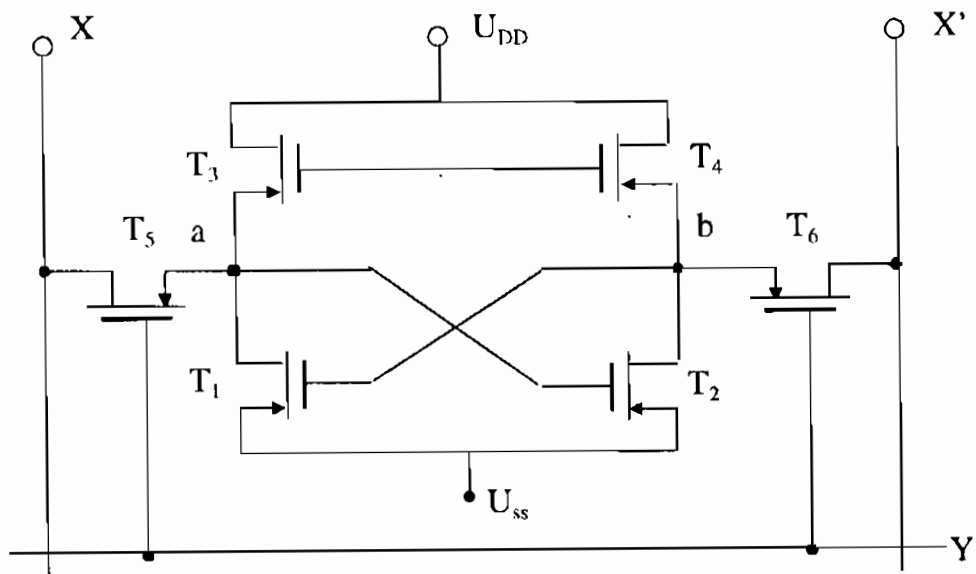
2.1. Loại tĩnh (SRAM)

+ Đơn vị nhớ:

Đơn vị nhớ của loại RAM tĩnh dùng MOSFET cũng là FF nhưng hơi khác dạng đã đề cập ở trên (hình 9.2).

Ta vẫn có hai đường XX' và đường Y để xác định từ và các bit của từ. FF được tạo thành bởi các transistor T_1 và T_2 , còn T_3 và T_4 hoạt động như hai điện trở. T_5 và T_6 làm việc như hai khoá điện tử để đọc ra hay viết vào FF đã được chọn.

Nếu tất cả các transistor đều là loại P thì U_{DD} thường âm còn U_{SS} có thể bằng 0V. Các đường XX' thường có điện thế âm.



Hình 9.2: Đơn vị nhớ RAM tĩnh dùng MOSFET

+ *Thao tác đọc:*

Muốn đọc, ta cho xung vào để làm giảm điện thế đường Y xuống âm, T_5 và T_6 dẫn. Giả sử T_2 dẫn thì cực máng của nó (nút b nối vào T_6) ở điện thế thấp, vì vậy T_6 sẽ dẫn một dòng điện chảy vào X' tạo ra dữ liệu. Trong khi đó, vì T_1 ngưng nên cực máng của nó ở điện thế âm cao bằng điện thế đường T_5 không dẫn.

+ *Thao tác viết:*

Muốn viết, ta cũng giảm điện thế đường Y xuống để mở T_5 và T_6 (chọn từ) rồi sau đó nối đường X hoặc X' xuống 0V tùy theo ta muốn T_1 hay T_2 ngưng dẫn. Nếu T_2 đang dẫn, nút b ở điện thế thấp gần 0V, nên khi nối đường X' xuống 0V sẽ không thay đổi gì cả. Nhưng nút a đang ở điện thế âm nhiều nên nếu nối X xuống 0V thì sẽ làm hạ điện thế tại đây xuống dưới trị số ngưỡng khiến T_2 ngưng dẫn làm T_1 trở nên dẫn.

+ *Đặc tính:*

- Người ta gọi loại bộ nhớ này là tĩnh bởi vì dữ liệu được lưu trữ vĩnh viễn trong mạch chùng nào mà ta cung cấp nguồn điện cho nó. Các đường điều khiển không cần thêm một xung nào.

- Loại bộ nhớ này tương đối rẻ hơn loại dùng transistor lưỡng cực vì hai lý do: Một là phương pháp chế tạo đơn giản; hai là kích thước của một đơn vị nhớ nhỏ nên có thể cho phép chế tạo bộ nhớ với dung lượng lớn trên một diện tích miếng silic nhỏ.

- Thời gian thâm nhập lớn hơn loại lưỡng cực vì có điện dung ký sinh ở cực cổng của MOSFET.

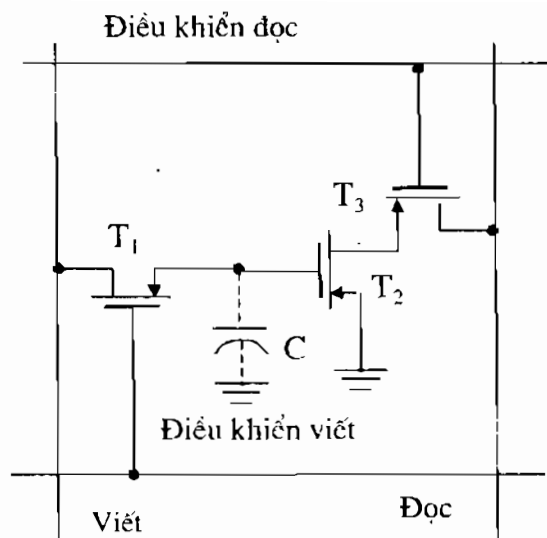
- Có thể chế tạo theo phương pháp kênh P, kênh N hay CMOS.

2.2. Loại động (DRAM)

+ *Đơn vị nhớ:*

Với trở kháng vào vô cùng lớn của MOSFET, người ta có thể chế tạo một loại đơn vị nhớ khác, sử dụng sự có mặt hay không có mặt của điện tích trên cực cổng của MOSFET để lưu trữ thông tin trong một khoảng thời gian nhất định, gọi là RAM động dùng MOSFET. Trên hình 9.3 là một dạng thông dụng của loại bộ nhớ này.

Mạch sử dụng 3 MOSFET nhưng không nối theo kiểu FF. T_1 và T_3 hoạt động như các khoá điện tử, T_2 có tác dụng như một mạch đảo. Dữ liệu được lưu trữ dưới dạng điện tích tụ điện cực cổng của T_2 .



Hình 9.3: Đơn vị nhớ RAM động dùng MOSFET

Đơn vị nhớ có 4 đường: đường điều khiển đọc và điều khiển viết thuộc về một từ (word), đường viết để ghi dữ liệu vào và đường đọc để lấy dữ liệu ra. Ta giả sử transistor là loại kênh P.

+ *Thao tác viết:*

Khi muốn viết, ta cho xung âm vào đường điều khiển viết để làm cho T_1 dẫn. Nếu đường viết ở 0V, tụ C không có điện tích, ta ghi bit 0. Nếu đường viết ở -U thì tụ C có điện tích và ta ghi bit 1.

+ *Thao tác đọc:*

Khi muốn đọc, ta cho xung âm vào đường điều khiển đọc và đường đọc. Nếu tụ C không có điện tích (bit 0) thì T_2 ngưng dẫn làm T_3 cũng ngưng dẫn nên trên đường đọc không có dòng điện ra. Ngược lại, nếu tụ C có điện tích thì T_2 dẫn làm T_3 cũng dẫn cho nên trên đường đọc có xung dòng điện chảy, ở mạch ngoài sẽ dò được và tạo ra một xung điện thế tương ứng.

Mạch có thể được làm cho đơn giản hơn bằng cách dùng một đường chung cho cả điều khiển đọc lẫn điều khiển viết. Lúc đó ta cần phải lưu ý rằng, xung điều khiển viết thường phải bé hơn xung điều khiển đọc để tránh làm cho (khi đọc) T_1 cũng dẫn điện khiến dữ liệu ở tụ C thay đổi.

Bởi vì mạch luôn luôn bị mất dần điện tích trên tụ C cho nên dữ liệu không lưu trữ được vĩnh viễn. Do đó, mạch phải được viết lại liên tục, người ta gọi đó là thao tác làm tươi (*refresh*) bộ nhớ. Muốn viết lại, trước hết ta phải đọc xem

tụ C có điện tích hay không rồi căn cứ vào đó mà viết lại như cũ. Chu kỳ viết lại thường vào khoảng 2ms.

+ *Đặc tính:*

- Khi không đọc-viết hay không viết lại, mạch hầu như không tiêu thụ công suất. Vì vậy, nói chung loại RAM này tiêu thụ ít công suất nhất so với các loại đã khảo sát ở trên.

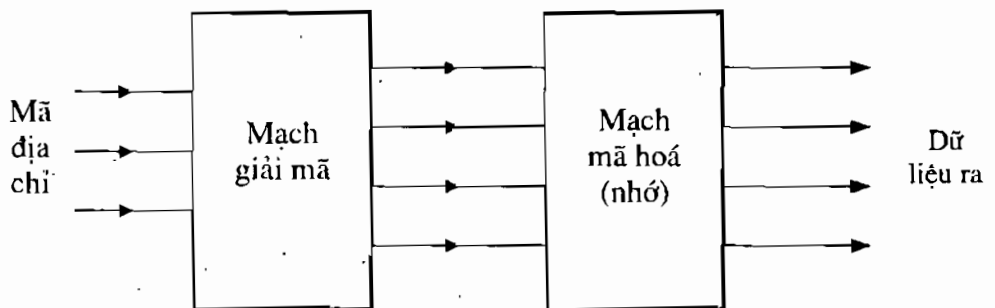
- Vì chỉ sử dụng ít transistor cho một đơn vị nhớ nên dễ chế tạo mạch với dung lượng lớn trên một diện tích bán dẫn nhỏ.

- Thời gian thâm nhập bé hơn các loại RAM tĩnh vì cấu trúc của mạch đơn giản hơn.

- Bộ nhớ loại này có một nhược điểm là luôn luôn cần phải làm tươi bộ nhớ khiến cho việc sử dụng trở nên rắc rối hơn và tín hiệu thường không phù hợp với TTL.

III. CÁC BỘ NHỚ ROM

Thuật ngữ ROM là viết tắt của từ tiếng Anh: *Read Only Memory*. Bộ nhớ ROM có đặc tính là chứa sẵn dữ liệu, khi cần ta chỉ đọc mà không viết ngay vào được. Dữ liệu trong bộ nhớ này đã được ghi vào từ trước và thường được lưu trữ một cách vĩnh viễn. Có loại ta không thể xóa được nội dung của nó và viết lại, có loại ta có thể xóa được nội dung hay viết lại nhưng cần phải qua một số thủ tục phức tạp (đây chính là các bộ nhớ cố định). Chính vì không cần viết ngay nên dạng mạch của loại này đơn giản hơn so với RAM vì không có các bộ phận cần cho thao tác viết vào.



Hình 9.4: Dạng cơ bản của ROM

Bộ nhớ ROM ngày càng thông dụng, nhất là loại bộ nhớ bán dẫn. Cả hai công nghệ lưỡng cực và MOSFET đều được sử dụng để chế tạo ROM. Thời

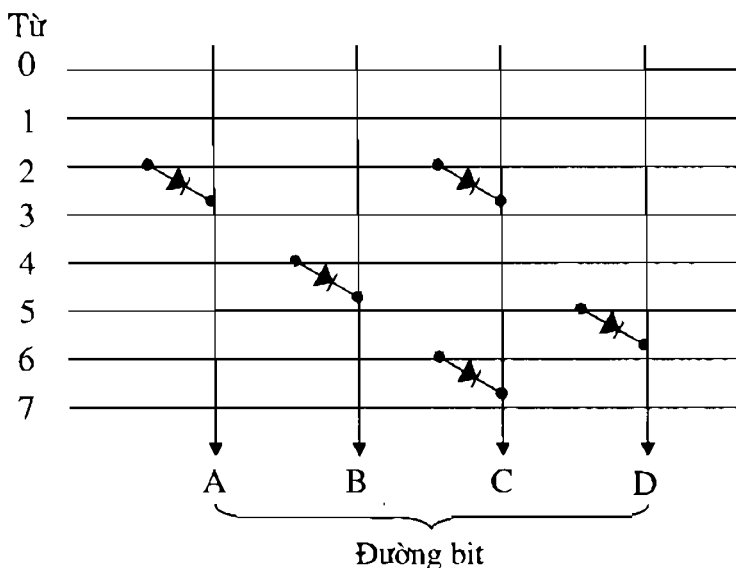
gian thâm nhập (at) của bộ nhớ lưỡng cực vào khoảng 50-90ns, còn đối với MOSFET thì gấp 10 lần (chậm hơn). ROM lưỡng cực nhanh hơn và có khả năng kích tốt hơn, trong khi đó, bộ nhớ MOSFET có cùng dung lượng thì có kích thước nhỏ hơn và tiêu thụ công suất ít hơn.

Thực tế thì ROM chính là một mạch mã hoá, do đó dưới dạng căn bản thì ROM thường bao gồm cả mạch giải mã như trên hình 9.4.

Dạng được giới thiệu trên là dạng phẳng, trong đó mỗi địa chỉ vào xác định một đường của mạch giải mã, nghĩa là xác định một từ, còn đầu ra là các bit của từ. Với cấu trúc như vậy, mạch không khác gì một mạch mã hoá để chuyển từ mã hệ 10 sang một mã nào đó.

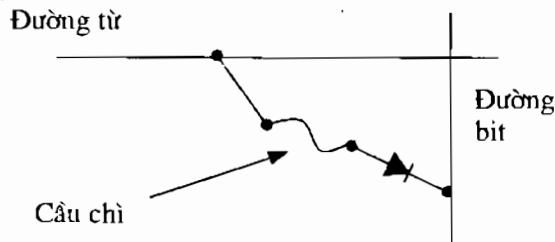
1. Bộ nhớ ROM dùng diode

Trước hết ta xét một ví dụ trên hình 9.5. Đây là một mạch nhớ ROM gồm 8 từ, mỗi từ dài 4 bit. Vị trí của một bit là giao điểm của hai đường là đường từ và đường bit. Tại giao điểm đó, nếu có 1 diode thì bit là 1, nếu không thì bit là 0. Muốn đọc thì ta cho địa chỉ, chẳng hạn 010 (2) vào mạch giải mã, tại đường của chữ số 2 sẽ có điện thế +U làm cho bit A và C bằng 1 (+U) còn bit B và D thì không có diode nên không có tín hiệu ra, tức là bằng 0. Như thế ở đường số 2 (hay địa chỉ 010) có dữ liệu 1010. Vậy, ta có thể mắc diode vào các giao điểm nào theo ý muốn và khi cần cũng có thể thay đổi vị trí các diode để "viết" dữ liệu khác vào mạch nhớ.



Hình 9.5: Bộ nhớ ROM dùng diode

Mạch như vậy rất đơn giản, gọn và khi không đọc thì hầu như không tiêu thụ năng lượng. Với dạng này chắc chắn chúng ta có thể chế tạo theo phương pháp tích hợp trên quy mô lớn (LSI), tức là làm toàn bộ mạch nhớ trên một miếng silic nhỏ. Thực tế, người ta đã chế tạo các bộ nhớ ROM dùng diode với dung lượng 4096 từ, mỗi từ dài 1 bit trên một miếng silic khoảng vài mm^2 và có at khoảng vài chục nanosec.

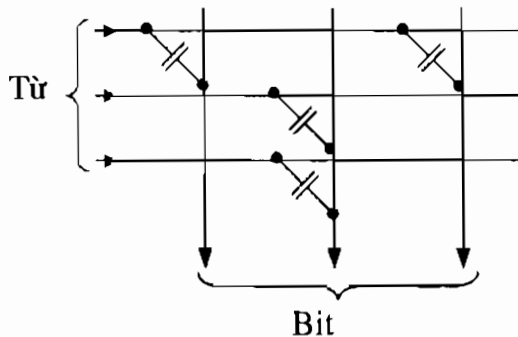


Hình 9.6: Nối cầu chì với diode ở PROM

Trong trường hợp bộ nhớ LSI kiểu này, người ta còn có thể nối tiếp mỗi diode với một cầu chì nhỏ ở tất cả mọi giao điểm như trên hình 9.6 để tạo ra một loại ROM lập trình được (*Programmable ROM* hay *PROM*). Khi sử dụng, ta viết dữ liệu vào bộ nhớ bằng cách phóng vào bộ nhớ một xung dòng điện có cường độ đủ để làm cháy cầu chì tại đó, như vậy còn cầu chì thì sẽ là bit 1. Với loại này nhà chế tạo có thể làm ra hàng loạt bộ nhớ LSI giống nhau và khi sử dụng, người sử dụng sẽ ghi vào theo ý muốn của mình. Dĩ nhiên rằng, sau khi viết xong ta không thể thay đổi nội dung được nữa.

2. Bộ nhớ ROM dùng tụ điện

Nếu thay đổi mạch nhớ ROM dùng diode bằng cách thay diode ở mỗi giao điểm bằng một tụ điện như trên hình 9.7 ta lại có một loại ROM khác là ROM dùng tụ điện.



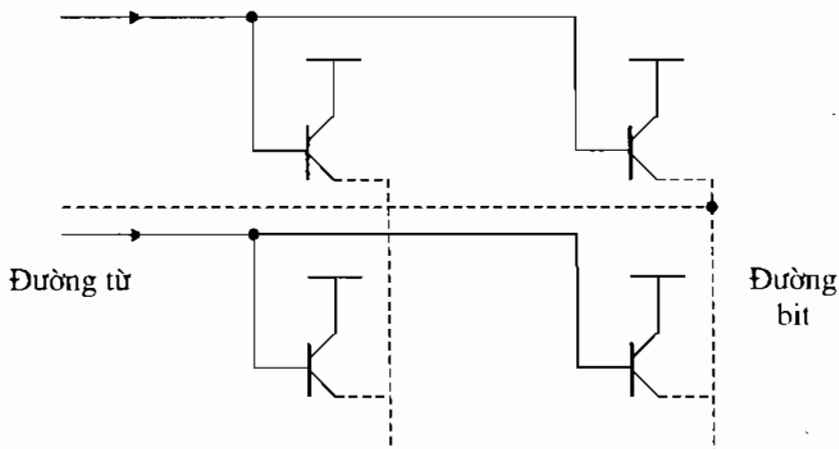
Hình 9.7: Bộ nhớ ROM dùng tụ điện

Khi cho một xung vào đường từ, tại giao điểm nào có tụ điện thì đường bit tương ứng sẽ có xung ra (bit 1).

Trong thực tế, người ta chế tạo loại bộ nhớ này bằng phương pháp mạch in. Trong đó các đường từ và bit nằm cách nhau bởi một lớp cách điện. Tại giao điểm có tụ điện thì ở đường từ và bit có thêm hai miếng kim loại mỏng, cách nhau bởi chất cách điện. Với cấu trúc như vậy ta có thể chế tạo các bộ nhớ với dung lượng tương đối lớn.

3. Bộ nhớ ROM dùng transistor lưỡng cực

Trong bộ nhớ ROM dùng transistor lưỡng cực là phần tử nằm ở giao điểm của đường từ và đường bit. Một mạng nhớ (ma trận nhớ) dùng transistor lưỡng cực tiêu biểu được biểu diễn như trên hình 9.8. Một transistor giao giữa đường từ và đường bit được tạo bởi đường nối với cực phát của transistor. Việc nối được hoàn thành trong quá trình chế tạo. Các bộ khuếch đại nhạy cảm nhận dòng điện ở cột được chọn để xác định mức logic là 0 hay là 1.



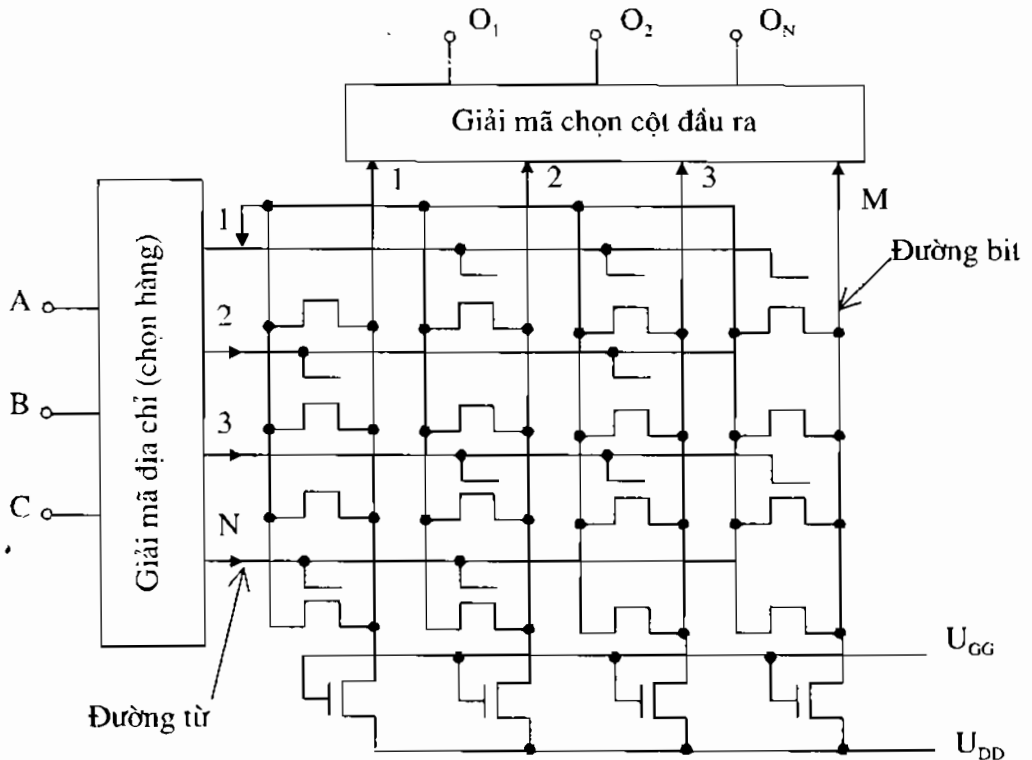
Hình 9.8: Bộ nhớ ROM dùng transistor lưỡng cực

4. Bộ nhớ ROM dùng MOSFET

MOSFET là công nghệ lý tưởng để chế tạo ROM nhờ mật độ linh kiện cao cho phép thiết kế các cấu trúc dạng ma trận chứa hàng nghìn phần tử MOSFET.

Trên hình 9.9 là sơ đồ một mạch ROM sử dụng MOSFET kênh P. Sự có mặt của một MOSFET nối các đường từ và bit được xác định bởi lớp ôxit mỏng thì ứng với một transistor. Trong quá trình chế tạo, nếu lớp ôxit đó dày thì xem như không có transistor, còn nếu lớp ôxit mỏng thì ứng với một transistor trong

ma trận được hoàn thành trong quá trình chế tạo mà người sử dụng không có thể can thiệp vào để thay đổi được.



Hình 9.9: Bộ nhớ ROM dùng MOSFET kênh P

5. Các bộ nhớ PROM, EPROM và EAROM

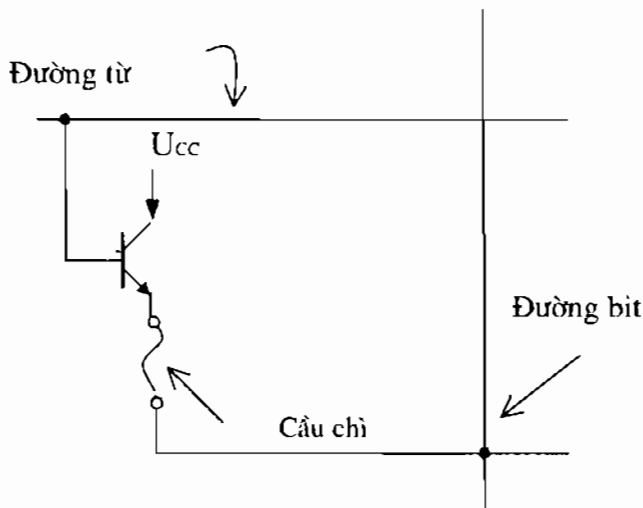
Các bộ nhớ PROM, EPROM và EAROM được viết bởi người sử dụng sau khi đã chế tạo.

- Bộ nhớ PROM chỉ có thể viết được một lần. Việc viết vào PROM yêu cầu phải có những mạch cung cấp dòng điện đủ lớn để đốt cháy cầu chì ở những bit mong muốn. Nhưng thiết bị viết này có thể do người sử dụng tự thiết kế hoặc mua sẵn.

- EPROM và EAROM mặc dù về giá thành thì đắt hơn PROM nhưng vì có thể viết nhiều lần nên tỏ ra thuận tiện trong việc thiết kế thử nghiệm hoặc sản xuất với khối lượng nhỏ. EPROM được xóa bằng tia cực tím còn EAROM thì được xóa bằng dòng điện, nên khi xóa có thể để trong mạch điện mà không phải gỡ ra như đối với EPROM.

5.1. Bộ nhớ PROM nối bằng cầu chì

Một bộ nhớ PROM (*Programmable ROM*) tiêu biểu đã được giới thiệu trên hình 9.6, trong đó chúng ta lưu ý đến sự có mặt của cầu chì nối tiếp với diode. Người ta cũng sản xuất loại PROM mà cầu chì lại được nối tiếp với một transistor (thay vì diode) như trên hình 9.10. Cầu chì thường được làm bằng Niken-Crome hay Silic đa tinh thể (*Poli-Crystalline Silicon*). Khi đường từ được chọn, transistor mở. Nếu có cầu chì thì đường bit được đưa lên điện thế 5V (U_{cc}). Nếu cầu chì đã bị làm đứt (trong quá trình viết vào) thì đường bit bị thả nổi.



Hình 9.10: Phần tử nhớ PROM nối bằng cầu chì

Bộ nhớ PROM được xác định bằng chương trình làm đứt cầu chì ở các phần tử giao nhau hay không. Đối với loại PROM có cầu chì chế tạo bằng đa tinh thể silic, xung làm đứt cầu chì có biên độ 20 - 30mA và độ rộng tăng dần cho đến khi cầu chì đứt.

5.2. Bộ nhớ EPROM có cửa thả nổi

Các bộ nhớ EPROM (*Erashle PROM*) có cửa thả nổi được sử dụng rất thuận tiện khi thiết kế các hệ thống số, đặc biệt là trong kỹ thuật vi xử lý. Vi mạch nhớ loại này đầu tiên do hãng Intel đưa ra năm 1971, đó là vi mạch 1702 có dung lượng 256 từ 8 bit sử dụng công nghệ PMOS và đóng trong vỏ 24 chân.

EPROM được thiết kế với các ô nhớ có thể xóa và viết lại được. Hiện nay EPROM đã được chế tạo dưới nhiều dạng với dung lượng khác nhau.

5.3. Bộ nhớ EAROM

Bộ nhớ EAROM (*Electrically Alterable ROM*) thay đổi nội dung được nhờ các tín hiệu điện tương tự như đối với bộ nhớ RAM. Chúng có thời gian viết dài và xoá từng ô nhớ riêng biệt, chẳng hạn vi mạch ER 3400 của hãng General Instruments có dung lượng 1024 từ 4 bit, thời gian viết là 1ms, thời gian xoá là 10ms và thời gian thâm nhập là 900ns.

EAROM thường được sử dụng trong những ứng dụng đòi hỏi không những bộ nhớ bất biến mà nội dung còn có thể thay đổi khi cần thiết. Nhược điểm quan trọng của các bộ nhớ EAROM là cần số lượng các mức điện áp nuôi khác nhau khi xoá, viết và đọc.

Câu hỏi và bài tập

1. Trình bày những khái niệm về bộ nhớ.
2. Phân loại các bộ nhớ bán dẫn đọc/viết.
3. Nêu cấu trúc của các loại bộ nhớ bán dẫn đọc/viết và hoạt động của nó.
4. Trình bày các đặc tính của các loại bộ nhớ bán dẫn đọc/viết.
5. Phân loại các bộ nhớ bán dẫn chỉ đọc.
6. Nêu cấu trúc của các loại bộ nhớ bán dẫn chỉ đọc và hoạt động của nó.
7. Trình bày các đặc tính của các loại bộ nhớ bán dẫn chỉ đọc.

Chương 10

CHUYỂN ĐỔI TÍN HIỆU

Mục tiêu

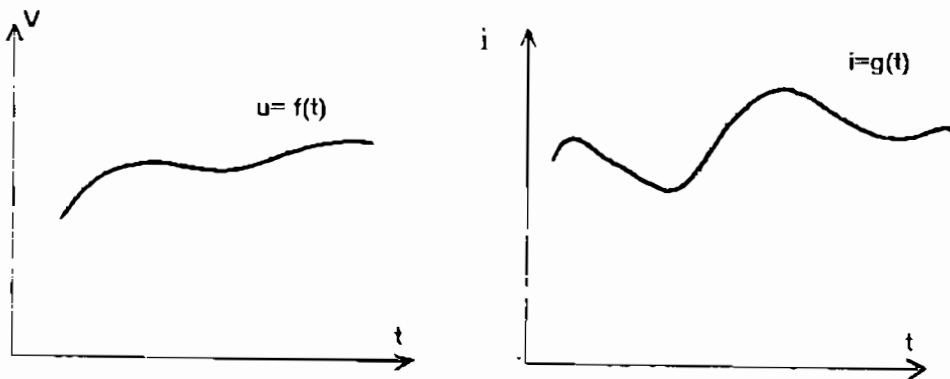
- Hiểu được các khái niệm về tín hiệu tương tự và tín hiệu số; sự cần thiết của việc chuyển đổi giữa các tín hiệu.
- Phân tích được hoạt động của các bộ chuyển đổi tín hiệu tương tự sang số và số sang tương tự.
- Hình thành được khả năng phân tích các mạch số và tương tự.

Nội dung trọng tâm

- Khái niệm về tín hiệu tương tự và tín hiệu số.
- Sơ đồ và hoạt động của các bộ chuyển đổi tín hiệu số sang tương tự.
- Sơ đồ và hoạt động của các bộ chuyển đổi tín hiệu tương tự sang số.

I. KHÁI NIỆM VỀ TÍN HIỆU TƯƠNG TỰ VÀ TÍN HIỆU SỐ

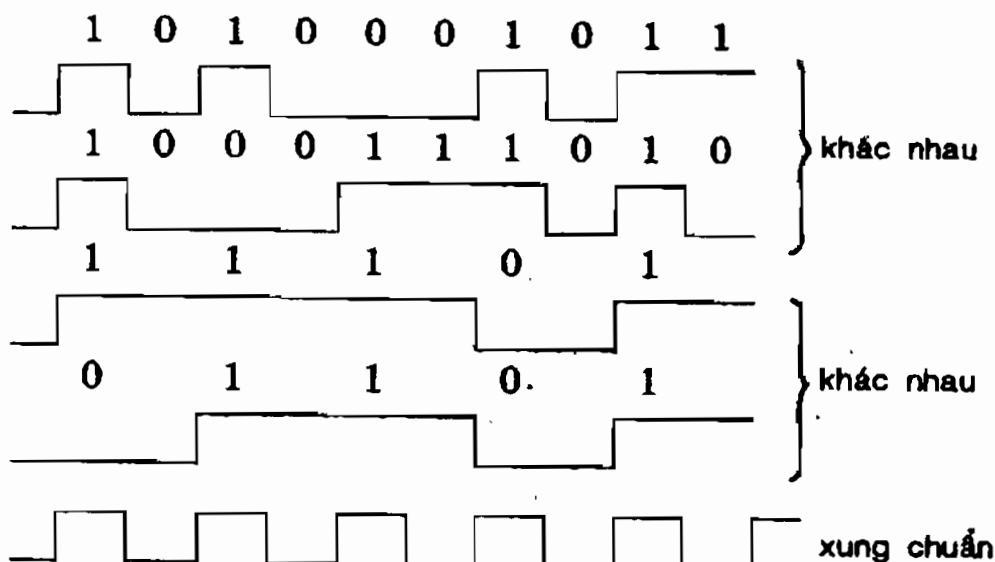
Tín hiệu tương tự (analog) thường là điện thế hay dòng điện có dạng biến thiên liên tục theo thời gian như biểu diễn trên hình 10.1. Thông tin mà các tín hiệu tương tự muốn truyền đạt cho con người là biên độ của nó. Biên độ này là một hàm của thời gian: $U=f(t)$; $I=g(t)$.



Hình 10.1: Tín hiệu tương tự (điện thế và dòng điện)

Tùy theo dạng của f và g mà ta có thể kết luận được về bản chất của tín hiệu u , i nói trên.

Khác với tín hiệu tương tự đã nói trên, tín hiệu số (digital) chứa thông tin ở vị trí các xung hay sự thay đổi đột ngột của biên độ (so với tín hiệu thời gian chuẩn) còn trị số tuyệt đối của biên độ xung thì không quan trọng (Hình 10.2).



Hình 10.2: Tín hiệu số

Như vậy, bản chất của hai loại tín hiệu tương tự và số khác nhau. Tín hiệu tương tự thường gặp trong thực tế vì dễ tạo ra, hơn nữa tính chất liên tục theo thời gian của nó phù hợp với các hiện tượng vật lý thông thường. Còn tín hiệu số chỉ xuất hiện trong các thiết bị số thường được dùng như một hình thức trung gian thuận tiện cho việc đo lường và xử lý tín hiệu. Nó thường không xuất hiện trực tiếp trong các hiện tượng thực tế.

Do đó, cần phải có các mạch trung gian để chuyển đổi tín hiệu từ dạng tín hiệu số sang tương tự và ngược lại mà người ta thường gọi chúng là các mạch *chuyển đổi số - tương tự* và *chuyển đổi tương tự - số*.

II. MẠCH CHUYỂN ĐỔI TÍN HIỆU SỐ SANG TÍN HIỆU TƯƠNG TỰ

Mạch chuyển đổi số - tương tự (*Digital - Analog Converter: DAC*) là một mạch dùng để chuyển đổi các tín hiệu số thành tín hiệu tương tự.

Như đã đề cập ở trên, vai trò của xung rất quan trọng trong các tín hiệu số. Mạch DAC cần phải biến đổi các nhóm xung khác nhau thành một mức độ điện thế hay một cường độ dòng điện tương ứng nào đó.

Người ta đã sử dụng ba phương pháp chính trong các mạch DAC là:

- Phương pháp tạo ra điện thế.
- Phương pháp tạo ra dòng điện.
- Phương pháp nhân.

1. DAC chế tạo theo phương pháp tạo ra điện thế

Trong loại DAC chế tạo theo phương pháp này có hai dạng:

- Dạng mạch với điện trở có trọng số khác nhau (*weighted resistor D/A converter*).
- Dạng mạch với điện trở hình thang (*R-2R ladder DAC*)

1.1. Mạch với điện trở có trọng số khác nhau

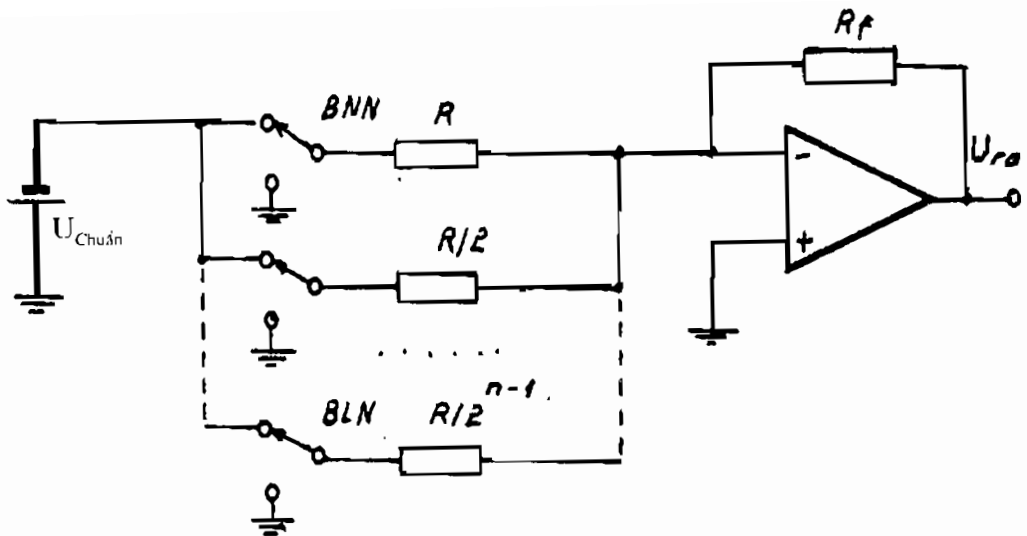
Trên hình 10.3 biểu diễn một mạch DAC dạng này. Mạch gồm có một nguồn điện thế chuẩn, các chuyển mạch, nhiều điện trở (mà trong trường hợp trên hình là n điện trở) có giá trị lần lượt là $R, R/2, R/4, \dots, R/2^{n-1}$ và một mạch khuếch đại thuật toán (mà từ đây trở về sau ta ký hiệu là KĐTT).

Với mạch như trên, khi một khoá điện nào được nối với nguồn điện thế chuẩn $U_{\text{chuẩn}}$ thì sẽ cung cấp cho mạch KĐTT một dòng điện có cường độ:

$$I_i = \frac{U_{\text{chuẩn}}}{R/2^i} \quad (0 \leq i \leq n-1)$$

Cường độ dòng điện này độc lập với các khoá còn lại.

Trong trường hợp có nhiều khoá điện nối với $U_{\text{chuẩn}}$ ta sẽ có nhiều dòng điện cộng chung lại chảy qua R_f tạo thành điện thế U_{ra} . Có thể thấy ngay rằng, biên độ U_{ra} phụ thuộc vào chỗ khoá điện nào được nối vào với $U_{\text{chuẩn}}$, tức là phụ thuộc vào giá trị của bit tương ứng trong tín hiệu số được đưa vào mạch chuyển đổi.



Hình 10.3: DAC với điện trở có trọng số khác nhau

Một cách tổng quát, với một DAC n bit (từ B^0 đến B^{n-1}) chế tạo theo phương pháp điện trở có trọng số khác nhau như sơ đồ trên hình 10.3, ta có thể tính tín hiệu đầu ra U_{ra} theo công thức sau (trong đó B^0 đến B^{n-1} có giá trị 0 hoặc 1):

$$U_{ra} = -U_{chuẩn} \frac{R_f}{R} (2^{n-1}B^{n-1} + 2^{n-2}B^{n-2} + \dots + 2^1B^1 + 2^0B^0)$$

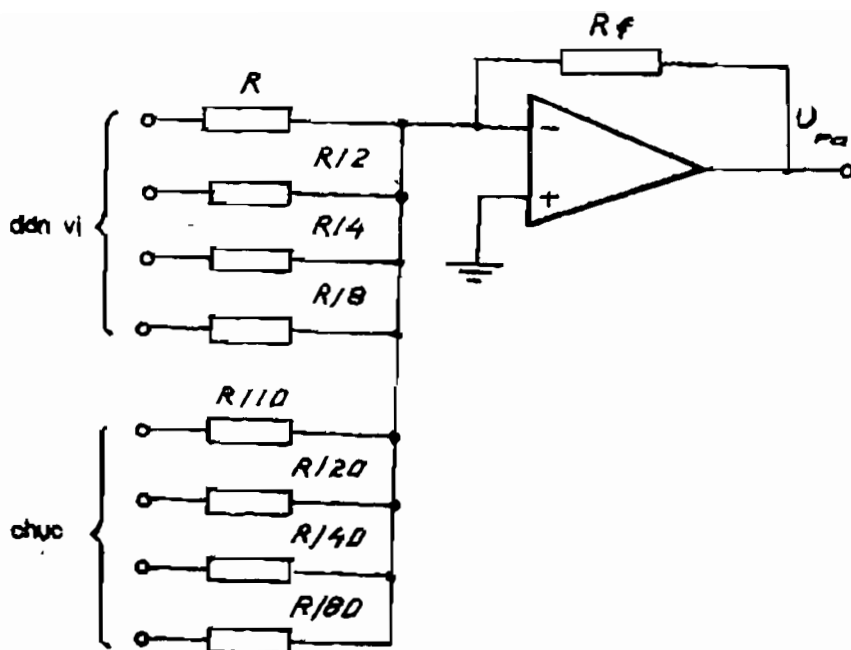
Ví dụ

Nếu đưa vào mạch chuyển đổi tín hiệu số là 1011, các khoá 1, 2 và 4 (tính từ bit nhỏ nhất) sẽ được nối vào $U_{chuẩn}$ và dòng điện tổng hợp sẽ là:

$$I_{th} = \frac{U_{chuẩn}}{R/8} + \frac{U_{chuẩn}}{R/8} = U_{chuẩn} \left(\frac{1}{R} + \frac{1}{R} + \frac{1}{R} \right) = \frac{U_{chuẩn}}{R}$$

Vậy tín hiệu ra là:

$$U_{ra} = -I_{th} \cdot R_f = -11 \cdot \frac{U_{chuẩn}}{R}$$



Hình 10.4: DAC với tín hiệu đầu vào là mã BCD 8421

Bây giờ nếu áp dụng công thức tổng quát đã nêu ta có:

$$U_{ra} = -U_{chuẩn} \cdot \frac{R_f}{R} (2^3 + 2^1 + 2^0) = -U_{chuẩn} \cdot \frac{R_f}{R} \cdot 11$$

Ta tìm lại được kết quả giống như trên.

Thành phần $U_{chuẩn} \cdot \frac{R_f}{R}$ là một hằng số cho nên điện thế đầu ra U_{ra} tỉ lệ với số 11 là giá trị hệ 10 của 1011 (hệ 2).

Trong trường hợp nếu sử dụng tín hiệu số dưới dạng mã BCD (chẳng hạn BCD 8421) thì các điện trở hợp thành từng nhóm biến điện trở một, mỗi nhóm có giá trị gấp 10 lần nhóm kế cận với nó. Trên hình 10.4 là một ví dụ với hai số hạng BCD là hàng đơn vị và hàng chục.

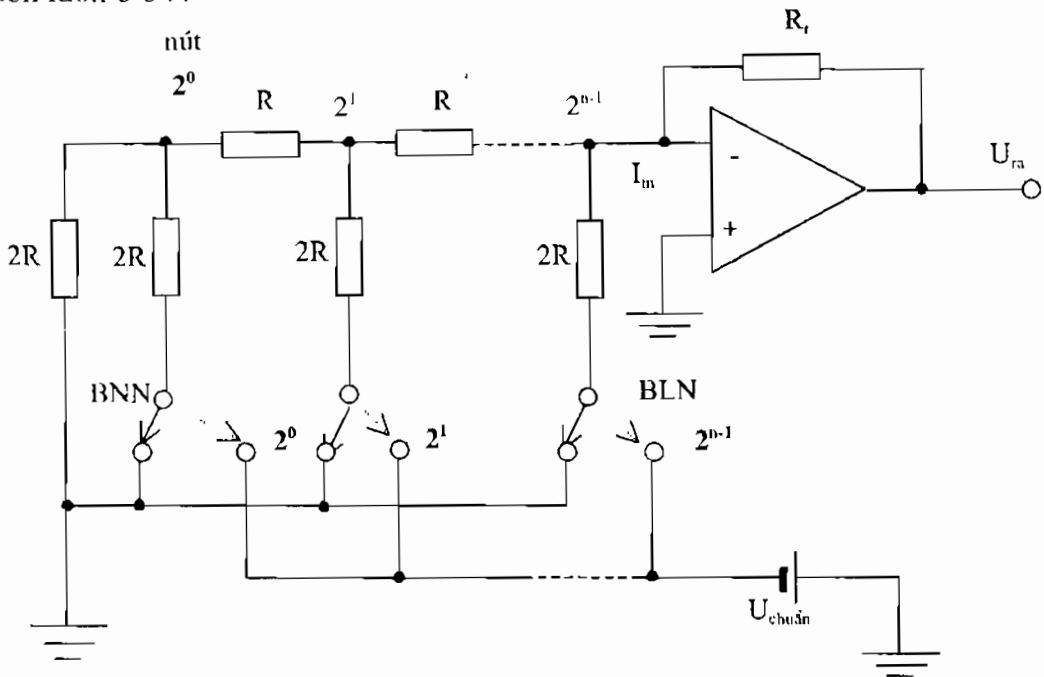
Mạch như trên có ưu điểm là đơn giản nhưng lại có khuyết điểm là độ chính xác và tính ổn định của kết quả phụ thuộc nhiều vào trị số tuyệt đối của các điện trở và khả năng biến thiên như nhau theo môi trường của các điện trở này. Vấn đề chế tạo các điện trở theo đúng tỉ lệ chính xác như vậy thường rất khó khăn và tốn kém. Ngoài ra, U_{ra} còn phụ thuộc vào cả độ chính xác và tính ổn định của nguồn $U_{chuẩn}$.

1.2. Mạch với điện trở hình thang

Loại mạch DAC với điện trở hình thang R - 2R giải quyết được một số khuyết điểm của loại trên. Trên hình 10.5 là sơ đồ của một mạch như vậy. Mạch gồm 2 loại điện trở R và 2R mắc theo hình thang với nhiều khoá điện (mỗi khoá cho một bit) và một nguồn điện thế chuẩn.

Để tính được U_{ra} theo sự đóng mở của các khoá điện, trước hết ta lưu ý một số giả thiết nhằm đơn giản hoá bài toán như sau:

- Giả sử nguồn $U_{chuẩn}$ có nội trở bằng 0 ôm. Như vậy, điện thế tương đương từ mỗi nút nhìn về phía trái đều là 2R.
- Tại nút 2^{n-1} do đặc tính của mạch KĐTT mà điện thế tại đây được coi như luôn luôn ở 0V.



Hình 10.5: Mạch DAC với điện trở hình thang

Như vậy, với các tính chất nói trên và vì mạch là một mạch tuyến tính nên ta có thể khảo sát ảnh hưởng của từng đầu vào đối với đầu ra U_{ra} .

Đại lượng cần tìm là I_{th} chảy vào mạch KĐTT khi có một số khoá điện nối vào nguồn $U_{chuẩn}$. Lúc đó ta có:

$$U_{ra} = - I_{th} \cdot R_f$$

Xét tại chuyển mạch tương ứng với bit thứ i (mà trên hình 10.5 ta ký hiệu là 2^i). Nút tương ứng trên mạch hình thang là nút 2^i . Sử dụng định lý Thevenin, khi đóng chuyển mạch vào $U_{\text{chuẩn}}/2$ và nguồn Thevenin có nội trở là R .

Từ đó ta thấy ngay rằng, tại nút 2^{i+1} (tiến về mạch KĐTT) ta sẽ có một nguồn tương đương Thevenin có nội trở là R và điện thế là $U_{\text{chuẩn}}/4$.

Từ những kết quả trên ta suy ra rằng, khi di chuyển về phía mạch KĐTT thì điện thế Thevenin tại mỗi nút bằng nửa trị số tại nút kề cận bên trái nó.

Như vậy, nếu từ nút 2^i đến nút 2^{n-2} có k nút, kể cả nút 2^i thì điện thế Thevenin tại nút 2^{n-2} do chuyển mạch 2^i là $U_{\text{chuẩn}}/2k$ và dòng điện tương ứng $U_{\text{chuẩn}}/2k \cdot 2R$ sẽ chảy tới mạch KĐTT.

Tóm lại, một cách tổng quát, ta có công thức để tính giá trị V_{ra} của mạch DAC n bit (từ B^0 đến B^{n-1}) với điện trở hình thang theo sơ đồ trên hình 10.5 như sau:

$$U_{ra} = -U_{\text{chuẩn}} \frac{R_f}{2^n R} (2^{n-1} B^{n-1} + 2^{n-2} B^{n-2} + \dots + 2^1 B^1 + 2^0 B^0)$$

Trong đó B^0 đến B^{n-1} có giá trị là 0 hoặc 1.

Ví dụ:

Với một mạch DAC 5 bit, ta đưa vào mạch tín hiệu 10101 hệ 2.

Vì bit nhỏ nhất bằng 1 nên chuyển mạch 2^0 nối với $U_{\text{chuẩn}}$ và dòng điện do nó cung cấp cho mạch KĐTT là:

$$I(2^0) = \frac{U_{\text{Chuẩn}}}{16} : \frac{1}{2R} = \frac{U_{\text{Chuẩn}}}{32R}$$

Bit kế tiếp bằng 0 nên chuyển mạch nối xuống đất và không tạo ra dòng điện.

Lập luận tương tự đối với các bit tiếp theo ta có:

$$I(2^2) = \frac{U_{\text{Chuẩn}}}{8R} = 4I(2^0)$$

$$I(2^4) = \frac{U_{\text{Chuẩn}}}{2R} = 16I(2^0)$$

Cuối cùng ta có thể tính được I_{th} như sau:

$$I_{th} = I(2^0) + I(2^2) + I(2^4) = U_{\text{chuẩn}} \left(\frac{1}{32R} + \frac{1}{8R} + \frac{1}{2R} \right)$$

Từ đó suy ra:

$$U_{ra} = R_f U_{\text{chuẩn}} \left(\frac{1}{32R} + \frac{1}{8R} + \frac{1}{2R} \right) = - R_f U_{\text{chuẩn}} \left(\frac{21}{32R} \right)$$

Bây giờ nếu áp dụng công thức tổng quát đã nêu ta có:

$$U_{ra} = - U_{chuẩn} \frac{R_f}{2^5 R} (2^4 + 2^2 + 2^0) = - R_f U_{chuẩn} \left(\frac{21}{32R} \right)$$

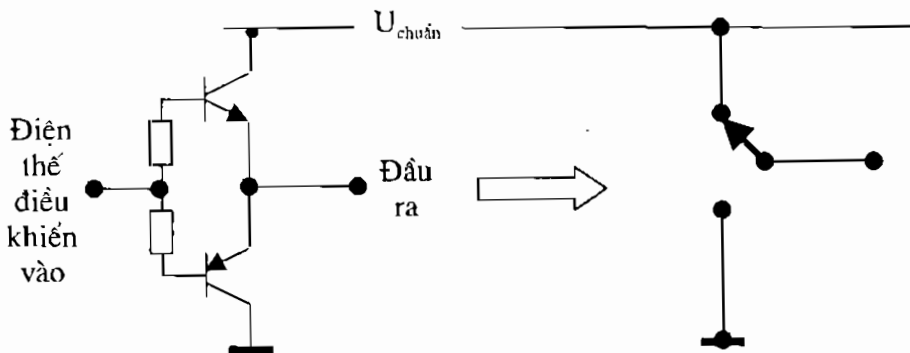
Ta tìm lại được kết quả như trên.

Như vậy, biên độ của tín hiệu ra U_{ra} tỉ lệ với giá trị hệ 2 của tín hiệu số đưa vào đúng như mong muốn của ta.

Các bộ biến đổi số - tương tự với điện trở hình thang dùng nhiều điện trở hơn so với loại dùng điện trở với trọng số khác nhau cho cùng một bit, nghĩa là tốn kém hơn, nhưng bù lại nó không rắc rối vì chỉ cần các điện trở có trị số là R và $2R$, nghĩa là với một tỉ lệ đơn giản, nên độ chính xác và tính ổn định của tín hiệu đầu ra U_{ra} được đảm bảo tốt hơn.

1.3. Chuyển mạch dùng cho DAC

Các DAC mô tả ở trên đều dùng chuyển mạch ở đầu vào. Loại chuyển mạch này có hai cực để có thể nối đầu vào với $U_{chuẩn}$ hay với $0V$. Muốn cho mạch có thể hoạt động bảo đảm và lên đến các tần số cao, thông thường các chuyển mạch này phải là rơle điện từ hay cơ khí. Trên hình 10.6 là sơ đồ một chuyển mạch thông dụng.



Hình 10.6: Sơ đồ chuyển mạch dùng transistor

Tuỳ theo điện thế điều khiển mà transistor npn hay pnp sẽ bão hoà và nối đầu ra lên điện thế chuẩn $U_{chuẩn}$ hay xuống masse ($0V$). Mạch này có ưu điểm là hoạt động giống như một khoá điện lý tưởng vì điện trở trong của transistor ở trạng thái bão hoà rất thấp và có giá trị đối xứng cho cả hai vị trí của khoá. Tất nhiên, đầu ra cũng chịu ảnh hưởng của U_{CE} bão hoà, mặc dù rất bé.

Người ta cũng sử dụng MOSFET để làm chuyển mạch với sơ đồ tương tự như trên. Loại này có ưu điểm là không bị ảnh hưởng về điện thế (như U_{CE}

bảo hoà) nhưng lại có nhược điểm là điện trở trong hơi lớn hơn loại transistor lưỡng cực.

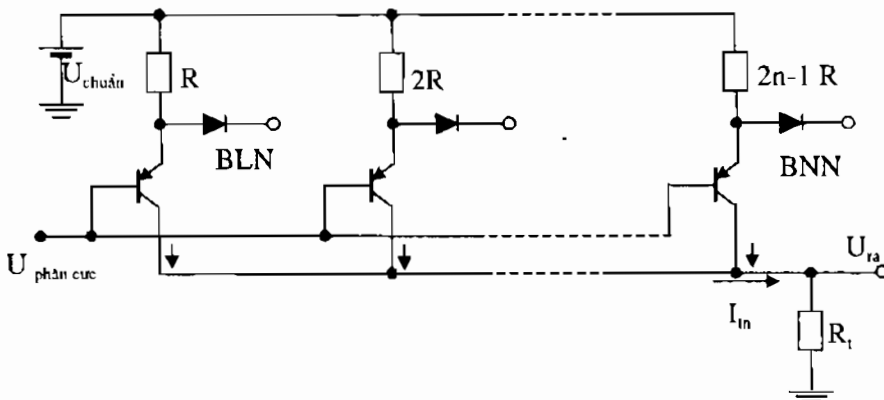
2. DAC chế tạo theo phương pháp tạo ra dòng điện

Các mạch DAC vừa khảo sát ở trên có đặc điểm là sử dụng mạch KĐTT ở phần cuối để tạo thành tín hiệu dưới dạng điện thế cho nên bị giới hạn bởi các đặc tính của mạch KĐTT, chẳng hạn vì tốc độ tác động chậm nên mạch không làm việc được ở tần số cao. Ngoài ra, trong nhiều ứng dụng ta lại cần một dòng điện đầu ra ở cường độ tỉ lệ với tín hiệu số ở đầu vào và cần làm việc ở tần số cao. Trong trường hợp này ta sẽ không sử dụng mạch KĐTT và có một loại DAC khác với các loại đã nêu trên.

Trên hình 10.7 là một sơ đồ đơn giản của loại DAC chế tạo theo phương pháp tạo ra dòng điện. Mỗi transistor và điện trở cực phát là một nguồn dòng điện. Cường độ của các nguồn dòng điện này thay đổi theo giá trị bit tương ứng của tín hiệu số đưa vào. Mỗi nguồn được điều khiển bởi bit tương ứng. Nếu bit có giá trị là 1 thì điện thế vào diode đủ dương làm cho diode ngưng dẫn, nguồn dòng hoạt động được và cung cấp một dòng điện cho mạch ngoài (R_i). Còn nếu bit có giá trị 0 thì điện thế vào diode đủ thấp làm cho diode dẫn nên transistor ngưng dẫn (vì điện thế cực phát thấp hơn cực gốc) khiến nguồn dòng điện không hoạt động.

Như vậy, bit nào bằng 1 sẽ cung cấp một nguồn dòng tương ứng cho mạch ngoài R_i . Tín hiệu ra lúc này có thể là dòng điện tổng hợp I_{th} (nếu ta bỏ R_i) hoặc là điện thế lấy ở đầu R_i .

Lưu ý rằng, R_i không nên có trị số lớn để U_{ra} khỏi ảnh hưởng đến các nguồn. Người ta thường dùng giá trị khoảng 100Ω với U_{ra} tối đa là 1 Volt.



Hình 10.7: Mạch DAC chế tạo theo phương pháp tạo ra dòng điện

Mạch khảo sát ở trên chỉ là một ví dụ đơn giản nên còn nhiều khuyết điểm, tuy nhiên nó cũng cho ta thấy được những nét chính của một mạch DAC chế tạo theo phương pháp tạo ra dòng điện.

Một ưu điểm dễ thấy của loại DAC này là có thể dễ dàng kết hợp các tín hiệu đầu ra bằng cách nối thẳng các đầu ra vào một điện trở R_L .

3. Các đặc tính quan trọng của DAC

Trong thực tế, khi sử dụng hay thiết kế một DAC người ta phải lưu ý đến các đặc tính quan trọng như sau:

3.1. Độ phân giải (*Solution*)

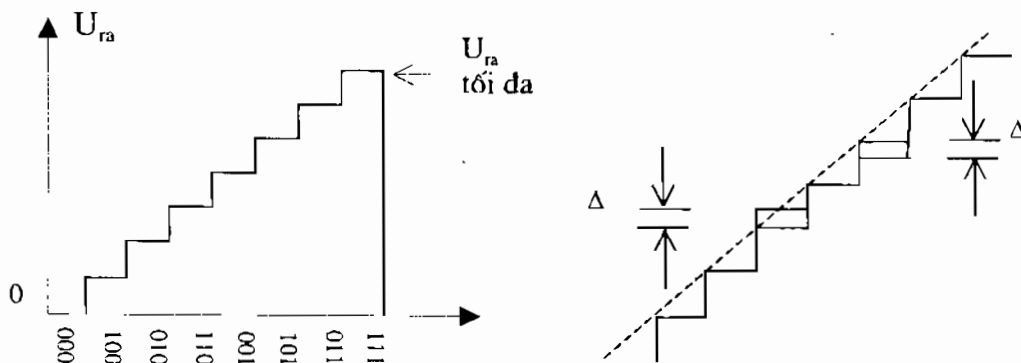
Độ phân giải liên quan đến số bit của một DAC. Nếu số bit là n thì số trạng thái của tín hiệu số nhị phân đưa vào là 2^n thì tín hiệu ra sẽ có 2^n mức điện thế khác nhau, do đó độ phân giải của mạch là $1/2^n$. Chẳng hạn, với một DAC 10 bit sẽ có $2^{10} = 1024$ mức điện thế khác nhau ở đầu ra và độ phân giải của mạch sẽ là $1/1024 \approx 0,1\%$.

Độ phân giải càng bé thì điện thế ở đầu ra có dạng càng liên tục và càng gần với thực tế.

3.2. Độ tuyến tính (*Linearity*)

Trong một DAC lý tưởng, sự tăng tín hiệu đầu vào sẽ tỷ lệ với sự tăng của tín hiệu tương tự ở đầu ra. Độ tuyến tính của một DAC phản ánh tính chính xác của yêu cầu này (nghĩa là DAC gần hay xa lý tưởng).

Để xác định độ tuyến tính của một DAC ta có thể làm như sau: Đo trị số tối đa của U_m (ứng với tất cả các bit số vào đều bằng 1) sau đó chia cho số mức mà theo lý thuyết sẽ có thể xuất hiện để tìm được chiều cao lý thuyết của mỗi mức. So sánh trị số tính toán này với trị số đo được của các mức trong thực tế. Nếu sự khác biệt không quá $1/2$ trị số lý thuyết thì DAC có thể xem là lý tưởng và chấp nhận được. Chẳng hạn với một DAC 3 bit có mức độ U_m tối đa là 3V (xem hình 10.8) thì chiều cao của một mức theo lý thuyết là $3V/(2^3-1) = 3V/7 \approx 0,43V$. Như vậy, nếu khác biệt giữa thực tế và lý thuyết không vượt quá $(1/2) \times 0,43V = 0,22V$ thì ta chấp nhận mạch DAC này. Lưu ý rằng, sự khác biệt có thể theo hướng lớn hơn hay bé hơn trị số lý tưởng.



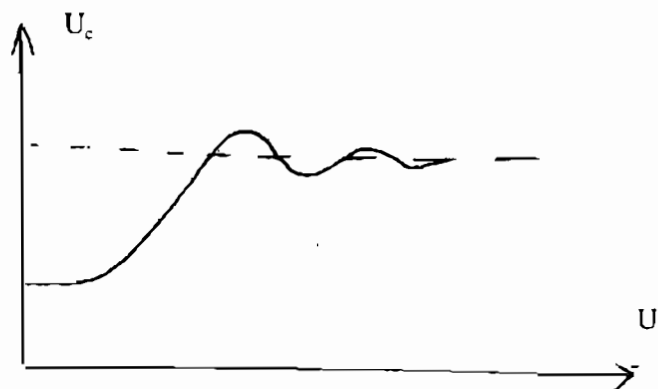
Hình 10.8: Minh họa độ tuyến tính của mạch DAC

3.3. Độ chính xác (Accuracy)

Độ chính xác của một DAC cho biết sự khác biệt giữa trị số thực tế của U_{ra} và trị số lý thuyết cho một giá trị bất kỳ của tín hiệu số ở đầu vào. Sự khác biệt càng bé thì độ chính xác càng cao.

3.4. Thời gian thiết lập (Settling time)

Khi tín hiệu số đầu vào của một mạch DAC thay đổi, tín hiệu đầu ra không thể thay đổi ngay lập tức mà phải qua một thời gian nào đó được gọi là thời gian thiết lập (Hình 10.9). Thời gian thiết lập phản ánh tính tác động nhanh của mạch, nó càng bé thì mạch hoạt động càng nhanh.



Hình 10.9: Thời gian thiết lập của DAC

3.5. Độ nhạy nhiệt (Temperature Sensitivity)

Với một đầu vào cố định, tín hiệu đầu ra của DAC thường thay đổi theo nhiệt độ, tính chất này được gọi là độ nhạy nhiệt của DAC.

Đối với các DAC thông thường thì giá trị cho phép của sự thay đổi này là $\pm 50 \text{ ppm}/^\circ\text{C}$ còn đối với các DAC chất lượng cao thì trị số này là $\pm 1,5 \text{ ppm}/^\circ\text{C}$.

III. MẠCH CHUYỂN ĐỔI TÍN HIỆU TƯƠNG TỰ SANG TÍN HIỆU SỐ

Trong phần này ta sẽ khảo sát mạch chuyển đổi thực hiện thao tác ngược với DAC đã nói ở trên. Đó là các mạch dùng để chuyển đổi các tín hiệu tương tự thành các tín hiệu số tương ứng (ở dạng nhị phân hay BCD) và gọi là các mạch chuyển đổi tương tự - số (*Analog-Digital Converter: ADC*).

Có nhiều phương pháp để thực hiện một mạch ADC nhưng ta chỉ khảo sát một vài phương pháp chính, trong đó đều có sử dụng một mạch đặc biệt gọi là mạch so sánh (*comparator*).

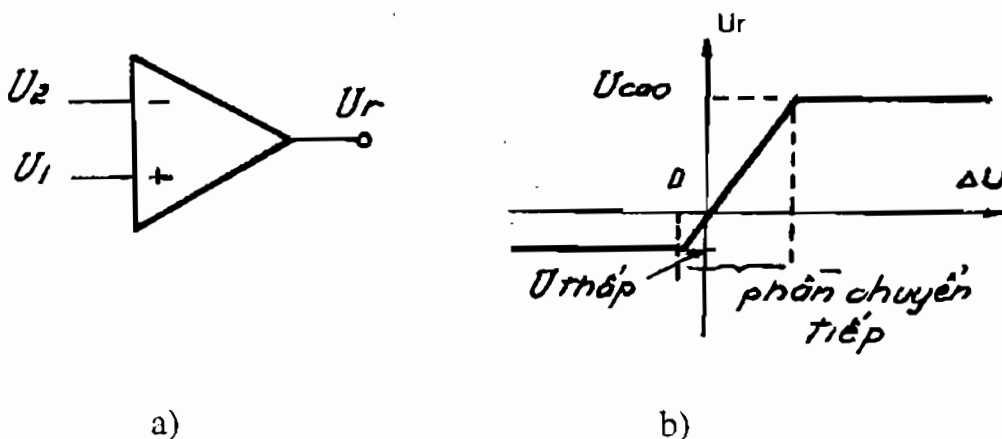
1. Mạch so sánh

Mạch so sánh là một mạch có hai đầu vào U_1 , U_2 : U_2 là đầu vào âm (-), U_1 là đầu vào dương (+) và một đầu ra U_r . Mạch được ký hiệu như trên hình 10.10.a và có đường đặc tính như trên hình 10.10.b.

Đầu ra U_r của mạch thường có hai trạng thái rõ rệt:

- Nếu $\Delta U = U_1 - U_2 > 0$ thì $U_r = U_{cao}$ (thường $> 0V$)
- Nếu $\Delta U = U_1 - U_2 < 0$ thì $U_r = U_{thấp}$ (thường $\leq 0V$)

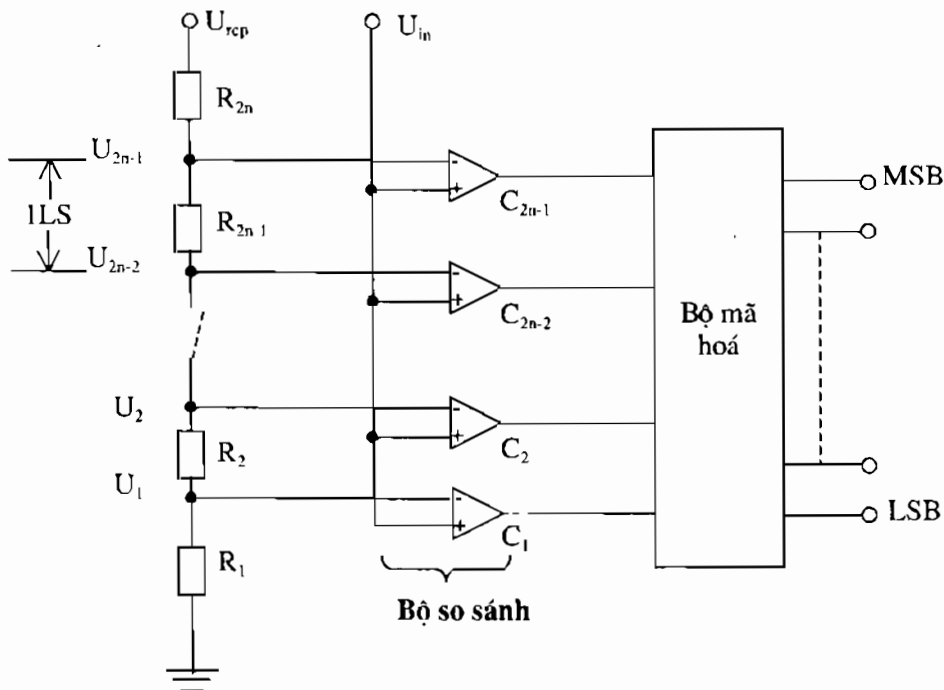
Khi ΔU gần bằng $0V$ ta có trạng thái chuyển tiếp (xem hình 10.11.b). Khoảng chuyển tiếp có bề rộng phụ thuộc vào dạng mạch thiết kế nhưng trị số này càng nhỏ thì sự so sánh càng chính xác. Hiện nay, người ta đã chế tạo các vi mạch chứa một hoặc vài bộ so sánh trong một vỏ rất tiện dụng.



Hình 10.10: Ký hiệu (a) và đặc tính (b) của mạch so sánh

2. Mạch ADC kiểu so sánh song song

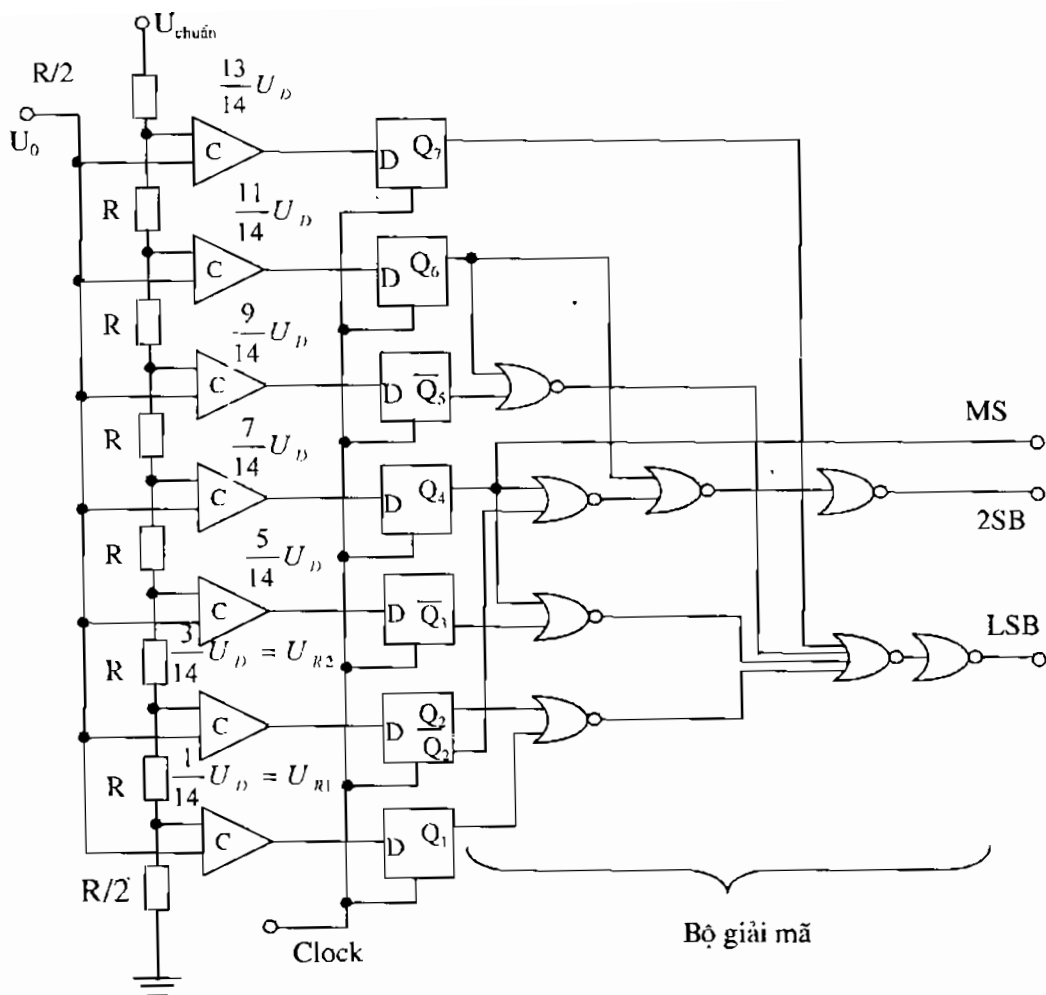
Trên hình 10.11 là sơ đồ mạch ADC kiểu so sánh song song (*parallel comparator type*). Mạch ADC kiểu này có tốc độ chuyển đổi cao nên còn được gọi là ADC nhanh (flash ADC).



Hình 10.11: Mạch ADC kiểu so sánh song song

Một mạch ADC n bit kiểu này cần $2^n - 1$ bộ so sánh (ký hiệu từ C_1 đến C_{2^n-1}) và 2^n điện trở (ký hiệu từ R_1 đến R_{2^n}), đầu ra của các bộ so sánh được đưa vào một bộ mã hoá ưu tiên thông qua một hệ thống nhớ đệm. Đầu ra của bộ mã hoá chính là đầu ra của ADC.

Để hiểu rõ hơn hoạt động của loại ADC này, ta xét ví dụ trên hình 10.12. Đó là sơ đồ của một ADC 3 bit. Điện áp vào thay đổi từ $0V$ đến U_0 . Đầu vào được chia thành 8 dải điện áp. Sáu trong tám dải có giá trị là $S = U_0/7$. Hai dải ở đầu cuối có giá trị $S/2 = U_0/14$. Khi đầu vào U_v có giá trị trong khoảng từ $0V$ đến $U_0/14$ thì đầu ra sẽ là 000 , khi U_v có giá trị từ $U_0/14$ đến $U_0/14 + S = 3U_0/14$ thì đầu ra tương ứng là 001 . Tương tự như vậy ta có thể xác định được giá trị các đầu ra đối với đầu vào bất kỳ (từ $0V$ đến U_0).



Hình 10.12: Mạch ADC 3 bit kiểu song song

3. Mạch ADC kiểu đếm

Trên hình 10.13 là sơ đồ của một mạch ADC kiểu đếm (*Counting ADC*). Mạch có những điểm chính như sau:

Mạch bao gồm các phần tử: một mạch so sánh, một FF - RS, một cổng AND để đóng-mở xung nhịp C_p từ bộ dao động chuẩn, một mạch đếm n bit và một mạch DAC n bit. Các phần tử này được mắc thành vòng hồi tiếp.

Lúc đầu, mạch đếm và FF được Reset nên điện áp ra U_{DAC} của DAC bằng 0V và cổng AND mở ra cho xung C_p vào mạch đếm. Lúc này điện áp vào U_v lớn hơn U_{DAC} (giả sử rằng cả hai đều dương) nên U_{ss} ở trị số thấp. U_{DAC} tăng

- Mạch DAC (và mạch đếm) càng có độ phân giải cao thì sự so sánh càng chính xác, vì chiều cao của bậc thang càng nhỏ.

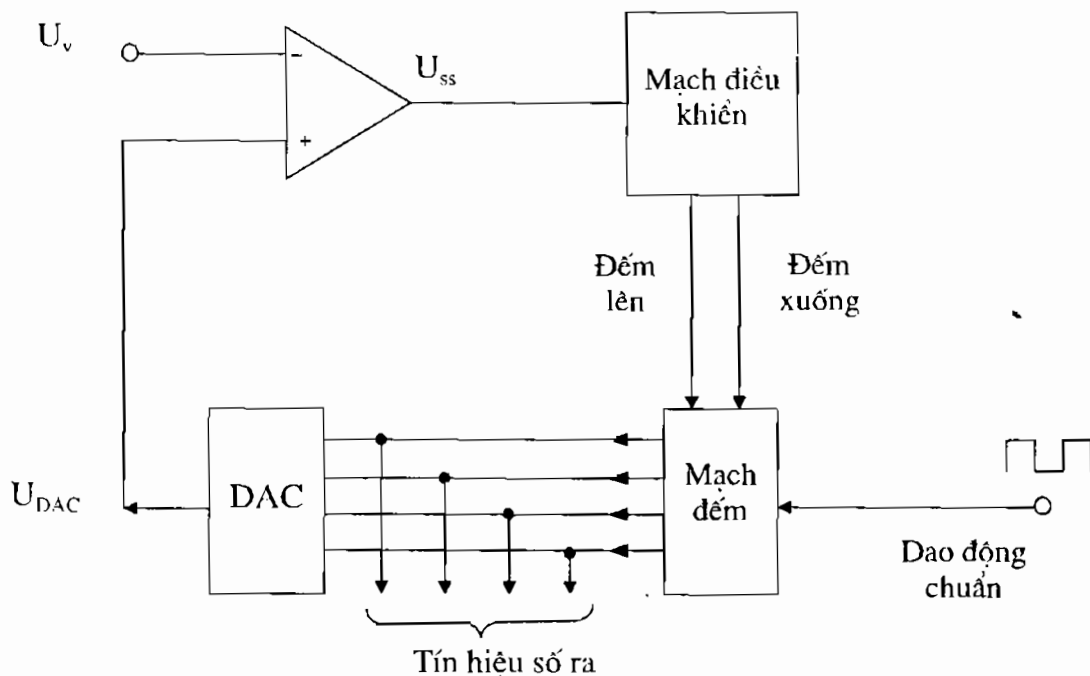
- Sự chuyển đổi luôn luôn cần một khoảng thời gian. Khoảng thời gian này tùy thuộc ở trị số của tín hiệu vào U_v và tần số của tín hiệu chuẩn (đối với cùng một DAC). U_v càng lớn thì thời gian chuyển đổi càng lâu, còn tần số C_p càng cao thì thời gian càng ngắn.

Cần lưu ý, sau mỗi lần chuyển đổi, các dữ liệu số thu được cần phải đưa vào một mạch lưu trữ hoặc chuyển đến bộ phận xử lý trước khi toàn mạch được Reset để không làm mất dữ liệu.

4. Mạch ADC so sánh liên tục

Trên hình 10.14 là sơ đồ của một mạch ADC chế tạo theo phương pháp so sánh liên tục. Mạch cho phép làm việc với tốc độ chuyển đổi nhanh hơn so với mạch đề cập ở trên, điều này có nghĩa là mạch cho phép chuyển đổi loại tín hiệu tương tự đầu vào U_x biến thiên nhanh hơn.

Đối với loại mạch này, khi sự khác biệt giữa U_v và U_{DAC} đến một trị số nào đó (do phân chuyển tiếp của mạch so sánh quyết định) thì U_{ss} sẽ ở một trong hai trị số U_{cao} (H) hoặc U_{thap} (L). Mạch điều khiển căn cứ vào trị số của U_{ss} để đưa tín hiệu đếm lên hay đếm xuống vào mạch đếm và do đó U_{DAC} sẽ biến đổi theo. Tuy nhiên U_{DAC} còn bị giới hạn bởi tần số của mạch dao động chuẩn nên chỉ có thể bám theo được những biến thiên của tín hiệu đầu vào đến một tốc độ giới hạn nào đó.

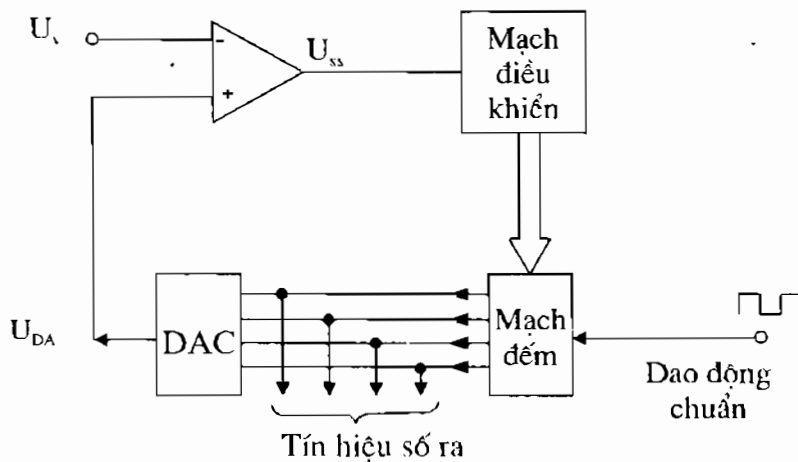


Hình 10.14: Mạch ADC so sánh liên tục

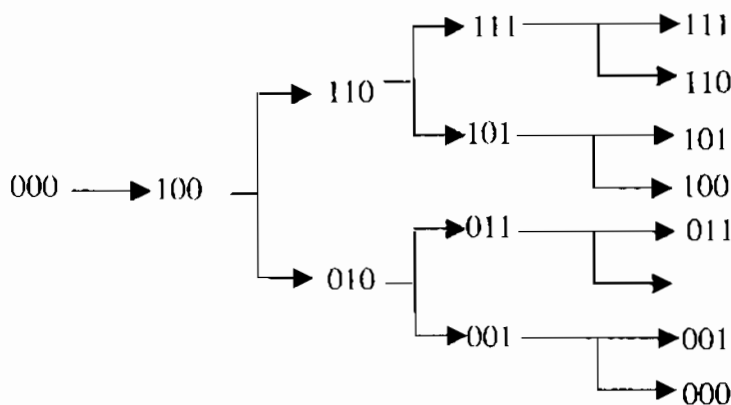
5. Mạch ADC xấp xỉ liên tiếp

Trên hình 10.15 là sơ đồ của một mạch ADC chế tạo theo phương pháp xấp xỉ liên tiếp (*successive approximation*). Mạch có dạng tương tự như sơ đồ mạch ADC so sánh liên tục (hình 10.14) nhưng ở đây ta sử dụng một mạch ghi thay vào vị trí mạch đếm. Mạch này cũng dùng để giảm thời gian chuyển đổi.

Khởi đầu mạch ghi ở trạng thái Reset (000...), sau đó mạch điều khiển kích thích để bit lớn xuất hiện. Có hai trường hợp xảy ra:



Hình 10.15: Mạch ADC xấp xỉ liên tiếp



Hình 10.16: Ví dụ minh họa hoạt động của ADC xấp xỉ liên tiếp 3 bit

- Nếu $U_{DAC} > U_v$ thì bit này bị bỏ đi.
- Nếu $U_{DAC} < U_v$ thì bit này được duy trì.

Sau đó bit kế tiếp xuất hiện để thử và cứ tiếp tục như vậy cho đến khi $U_{DAC} = U_v$ với một độ chính xác nào đó.

Trên hình 10.16 là ví dụ tiêu biểu với ADC 3 bit, là giản đồ hình cây cho thấy sự thay đổi các bit tương ứng với các U_v có giá trị khác nhau.

Với phương pháp này thời gian chuyển đổi dài nhất là $2n$ lần chu kỳ xung của dao động chuẩn đối với một mạch n bit. Trong khi đó, nếu so sánh với phương pháp bậc thang lên thì thời gian tối đa có thể là $2^n - 1$ lần chu kỳ xung chuẩn. Nhưng với mạch dạng này ta phải trả giá là mạch điều khiển rắc rối hơn.

6. Độ chính xác của mạch chuyển đổi tương tự - số

Do sự phức tạp của phương pháp chuyển đổi tín hiệu từ tương tự sang số mà các mạch ADC có độ chính xác không bằng các mạch DAC.

6.1. Sai số tĩnh

Khi thực hiện chuyển đổi các tín hiệu tương tự thành các tín hiệu số với số bit hữu hạn thường xuất hiện sai số hệ thống. Các sai số này được gọi là sai số lượng tử (*Quantization error*). Giá trị của sai số lượng tử bằng một nửa giá trị của điện áp U_{LSB} cần thiết thêm vào điện áp vào U_v để làm thay đổi một mã của bit ý nghĩa ít nhất (*Least Significant Bit: LSB*). Để giảm nhỏ tạp âm lượng tử thì cần phải giảm nhỏ U_{LSB} nghĩa là cần phải tăng số bit.

Bên cạnh sai số hệ thống do lượng tử hoá còn có sai số cũng đáng kể do kết cấu của mạch gây ra.

Các sai số do kết cấu của mạch gây ra càng được khắc phục để tạo ra các ADC gần như lý tưởng.

6.2. Sai số động

Khi ứng dụng các ADC, ta có thể phân thành hai nhóm lớn:

- ADC dùng trong các Von-kế số.
- ADC dùng trong xử lý tín hiệu.

Trong trường hợp đầu, tín hiệu đầu vào không đổi trong suốt thời gian chuyển đổi.

Còn trong trường hợp thứ hai thì ngược lại, tín hiệu đầu vào liên tục biến động. Vì vậy, người ta tiến hành lấy mẫu tín hiệu qua các khoảng thời gian bằng nhau, gọi là *chu kỳ lấy mẫu*. Các dữ liệu thu được sau đó sẽ được chuyển đổi thành dạng số nhờ các ADC. Lưu ý rằng, để có thể phục hồi được chính xác tín hiệu tương tự thì việc lấy mẫu cần phải thoả mãn định lý lấy mẫu (*định lý Shannon*) mà nội dung có thể tóm tắt như sau:

Tần số lấy mẫu f_s ít nhất phải lớn hơn 2 lần tần số lớn nhất của tín hiệu f_{max} .

Vì thế, thời gian chuyển đổi của ADC cần phải nhỏ hơn $1/2f_{max}$. Và khi đánh giá độ chính xác của ADC thì cần phải khảo sát kết hợp với các bộ lấy mẫu.

6.3. Các yếu tố ảnh hưởng đến độ chính xác của ADC

Xét dưới góc độ là độ chính xác thì bất cứ một ADC nào cũng gồm có hai phần là phần tạo tín hiệu so sánh và mạch so sánh.

+ *Phần tạo tín hiệu so sánh:*

- Nếu phần tạo tín hiệu so sánh là DAC (chẳng hạn như loại dùng tín hiệu bậc thang lên) thì độ phân giải, độ tuyến tính và độ chính xác của DAC này ảnh hưởng đến độ chính xác của toàn mạch.

- Nếu phần tạo tín hiệu so sánh là mạch tạo tín hiệu răng cưa thì sự ổn định của độ dốc và độ tuyến tính của tín hiệu này rất quan trọng.

+ *Phần mạch so sánh:*

Đối với mạch so sánh, chính khoảng chuyển tiếp của tín hiệu vào là một yếu tố gây ra sự kém chính xác, bởi vì tín hiệu vào phải khác nhau một lượng nào đó thì điện thế ra của mạch so sánh U_{ss} mới chuyển đổi trạng thái. Ngoài ra, khoảng chuyển tiếp này lại thay đổi theo trị số tuyệt đối của điện thế vào, tức là ứng với mỗi biên độ ta cần có một khoảng chuyển tiếp riêng, vì vậy, mà xảy ra hiện tượng không tuyến tính trong phép so sánh. Tùy theo sự thiết kế của mạch so sánh mà bề rộng chuyển tiếp còn có thay đổi ít nhiều theo nhiệt độ.

Ngoài các điểm chính nói trên, độ chính xác của mạch ADC cũng còn bị ảnh hưởng bởi độ ổn định của mạch dao động chuẩn và thời gian giao hoán của các cổng logic.

Để tăng độ chính xác, ngày nay người ta đã chế tạo nhiều mạch DAC và ADC bằng phương pháp tích hợp. Các phần tử của mạch có tính chất biến thiên theo nhiệt độ giống nhau, bảo đảm sự ổn định của kết quả chuyển đổi.

Câu hỏi và bài tập

1. Trình bày một số khái niệm về tín hiệu tương tự và tín hiệu số.
2. Phân tích sơ đồ và hoạt động của các bộ chuyển đổi tín hiệu số sang tương tự (DAC):
 - a) DAC chế tạo theo phương pháp tạo ra điện thế.
 - b) DAC chế tạo theo phương pháp tạo ra dòng điện.
 - c) DAC chế tạo theo phương pháp nhân.
3. Phân tích các đặc tính quan trọng của DAC.
4. Trình bày về mạch so sánh.
5. Phân tích sơ đồ và hoạt động của các bộ chuyển đổi tín hiệu tương tự sang số (ADC):
 - a) ADC kiểu so sánh song song.
 - b) ADC kiểu đếm.
 - c) ADC kiểu so sánh liên tục.
 - d) ADC kiểu xấp xỉ liên tiếp.
 - e) ADC kiểu điện áp răng cưa.
6. Trình bày về độ chính xác của mạch chuyển đổi tương tự sang số.

Chương 11

CÁC VI MẠCH SỐ THÔNG DỤNG VÀ MỘT SỐ SƠ ĐỒ THỰC TẾ

Mục tiêu

- Hiểu được cấu trúc, chức năng và nguyên lý hoạt động của một số vi mạch số thông dụng.
- Phân tích được hoạt động của một số mạch ứng dụng.
- Hình thành được kỹ năng sử dụng các vi mạch để ghép nối thành những mạch ứng dụng.

Nội dung trọng tâm

- Cấu trúc, chức năng, đặc điểm và hoạt động của một số vi mạch số thông dụng.
- Một số sơ đồ mạch thực tế được xây dựng từ các vi mạch số thông dụng.

I. CÁC VI MẠCH SỐ THÔNG DỤNG

1. Các vi mạch chứa các cổng logic

Mục này giới thiệu một số khối cổng mà ta thường gọi theo thói quen với cái tên ở dạng mạch tích hợp là “vi mạch”. Đó là các mạch tích hợp chứa một số cổng logic cơ bản trong một vỏ.

1.1. Khối cổng loại TTL/LS

1.1.1. 7400/74LS00 (hình 11.1.a)

Khối cổng này chứa 4 cổng NAND. Đó là một trong những khối cổng cơ sở để thiết kế các mạch số và rất dễ sử dụng. Có rất nhiều ứng dụng. Nguồn cung cấp $U_{cc} = +5V$.

1.1.2. 7402/74LS02 (hình 11.1.b)

Cũng là một vi mạch thông dụng nhưng được dùng ít hơn so với 7400/74LS00. Chứa 4 cổng NOR. Nguồn cung cấp $U_{cc} = +5V$.

1.1.3. 7404/74LS04 (hình 11.1.c)

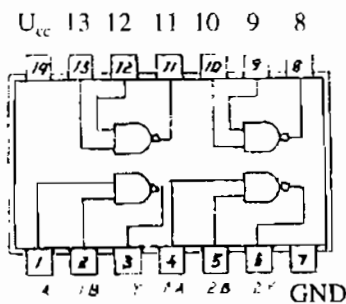
Chứa 6 bộ đảo trong một khối cổng. Rất quan trọng và được dùng trong hầu hết các sơ đồ mạch nhằm thay đổi một số hệ 2 thành phần bù của nó. Nghĩa là 1- $\rightarrow$ 0 và 0- $\rightarrow$ 1. Nguồn cung cấp $U_{cc} = +5V$.

1.1.4. 7408/74LS08 (hình 11.1.d)

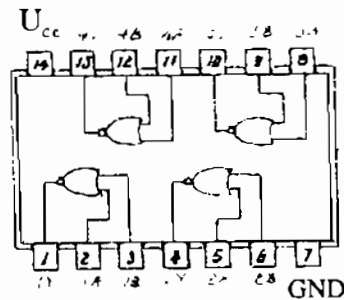
Chứa 4 cổng AND trong một khối cổng. Không thông dụng bằng 7400/74LS00. Nguồn cung cấp $U_{cc} = +5V$.

1.1.5. 7432/74LS32 (hình 11.1.e)

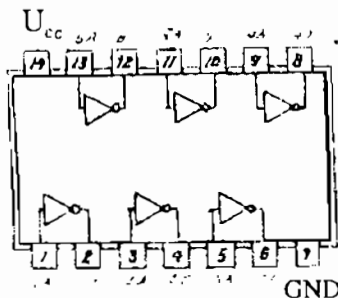
Chứa 4 cổng OR. Kém thông dụng hơn so với 7402/74LS02 nhưng lại rất thuận tiện để thiết kế các mạch chọn dữ liệu đơn giản. Nguồn cung cấp $U_{cc} = +5V$.



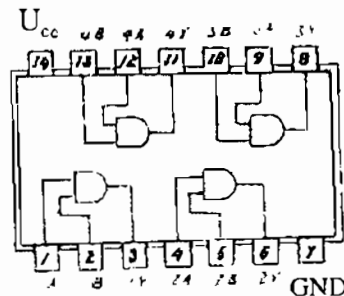
a) 7400



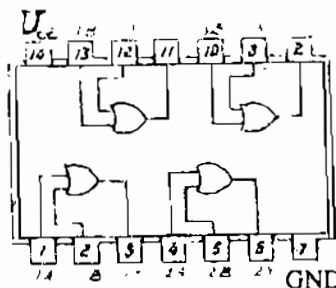
b) 7402



c) 7404



d) 7408



e) 7432

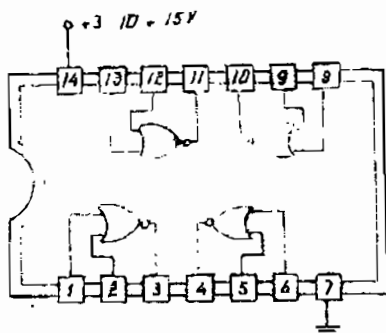
Hình 11.1: Một số vi mạch chứa cổng logic cơ bản họ TTL

1.2. Khối cổng loại CMOS

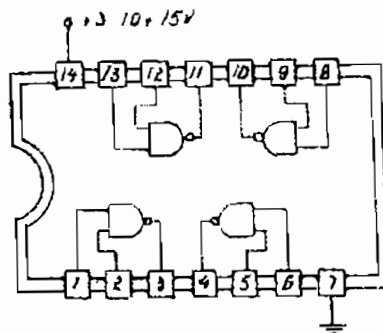
1.2.1. 4001 (hình 11.2.a)

Chứa 4 cổng NOR. Là vi mạch quan trọng trong thiết kế mạch số. Trở kháng đầu vào cao nên có nhiều ứng dụng hơn so với 7402/74LS02. Nguồn cung cấp $U_{DD} = +3V$ đến $+15V$.

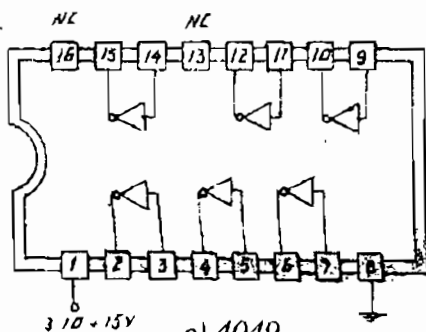
1.2.2. 4011 (hình 11.2.b)



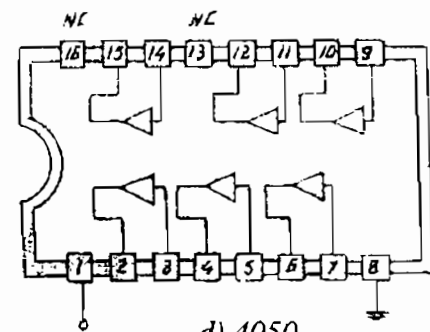
a) 4001



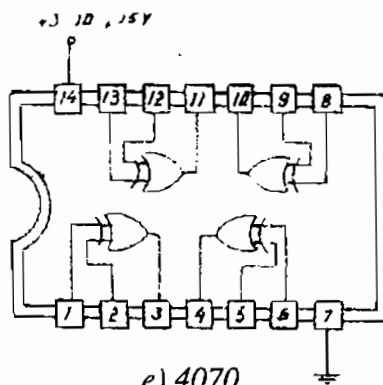
b) 4011



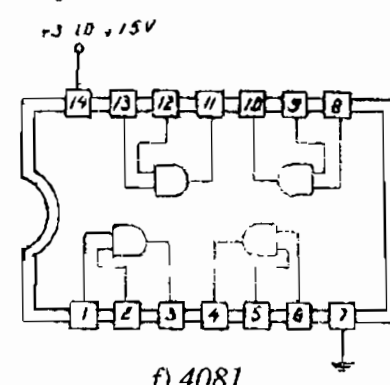
c) 4049



d) 4050



e) 4070



f) 4081

Hình 11.2: Một số vi mạch chứa cổng logic cơ bản họ CMOS

Chứa 4 cổng NAND. Cũng là vi mạch có nhiều ứng dụng trong thiết kế mạch số. Chức năng tương tự 7400/74LS00. Nguồn cung cấp $U_{DD} = +3V$ đến $+15V$.

1.2.3. 4049 (hình 11.2.c)

Chứa 6 bộ đảo trong một khối cổng. Ngoài các ứng dụng đảo tín hiệu logic và phối ghép CMOS - TTL, thường được dùng trong các bộ dao động và phát xung. Nguồn cung cấp $U_{DD} = +3V$ đến $+15V$.

1.2.4. 4050 (hình 11.2.d)

Chứa 6 bộ đệm không đảo. Là vi mạch cơ bản dùng trong phối ghép CMOS-TTL. Dòng cung cấp lớn hơn một ít so với các vi mạch CMOS tiêu chuẩn. Nguồn cung cấp $U_{DD} = +3V$ đến $+15V$.

1.2.5. 4070 (hình 11.2.e)

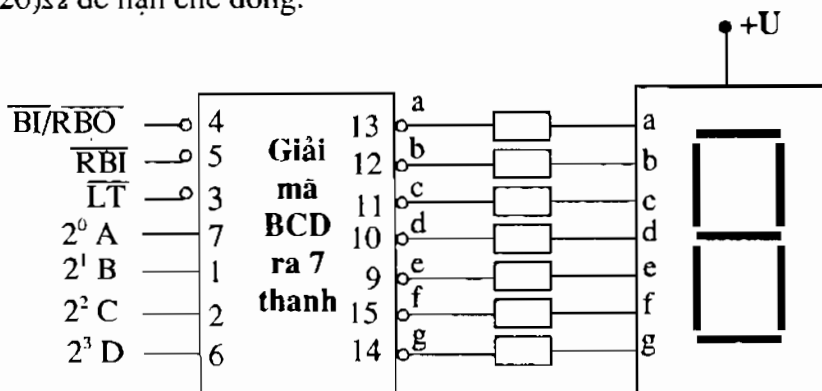
Chứa 4 cổng EXCLUSIVE-OR. Có nhiều ứng dụng như cộng nhị phân, so sánh các từ nhị phân v.v... Nguồn cung cấp $U_{DD} = +3V$ đến $+15V$.

1.2.6. 4081 (hình 11.2.f)

Chứa 4 cổng AND trong một khối. Thường được dùng để thiết kế các bộ đếm và các sơ đồ mạch logic. Tuy nhiên không thông dụng bằng 4011. Nguồn cung cấp $U_{DD} = +3V$ đến $+15V$.

2. Vi mạch giải mã BCD ra 7 thanh

IC 7446/7447 là các mạch giải mã BCD ra 7 thanh, họ TTL, các đầu ra là tích cực thấp. Hình 11.3 vẽ sơ đồ khối của IC 7446, 7447 và cách nối chúng với hiển thị 7 thanh loại anod chung (Anod Common:AC). Điện áp cung cấp cho hiển thị cũng chính là điện áp cung cấp cho IC giải mã ($U = 5V$). Điện trở $R = (150 \div 220)\Omega$ để hạn chế dòng.



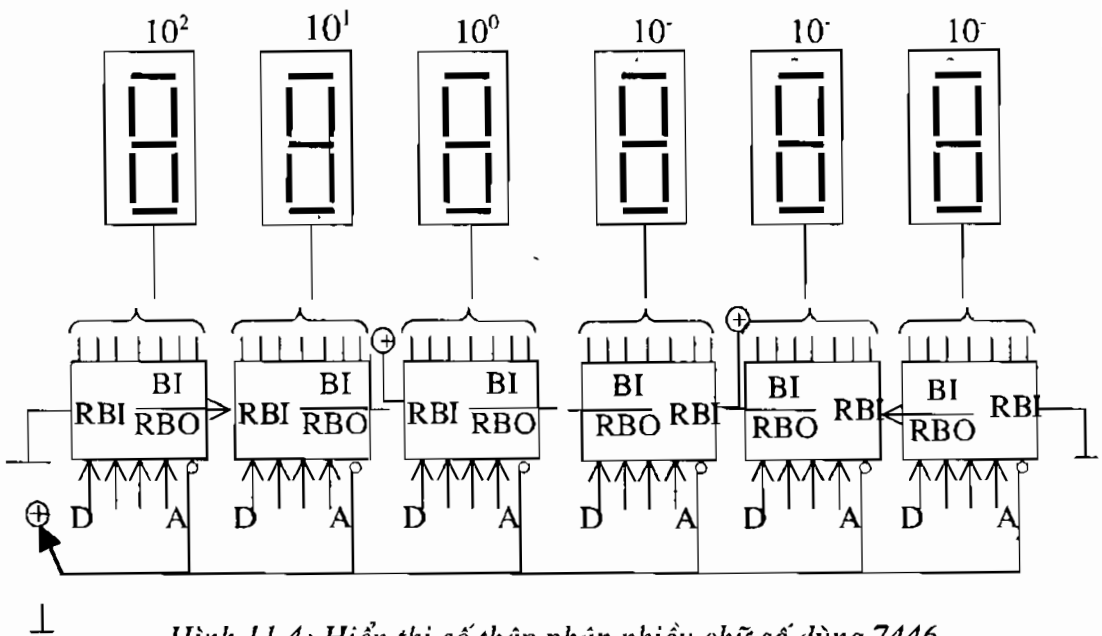
Hình 11.3: IC 7446,7447 và cách nối với hiển thị LED 7 thanh

Ta thấy ngoài 4 đầu vào D,C,B và A để đặt số thập phân, IC 7446, 7447 còn có 3 đầu vào đặc biệt là $\overline{LT}$, $\overline{RBI}$, $\overline{BI}$, $\overline{RBO}$. Chúng có các chức năng sau:

- $\overline{LT}$ (Lamp Test): Đầu vào thử đèn, tích cực mức thấp dùng để thử hiển thị 7 thanh. Khi đầu $\overline{BI}/\overline{RBO}$ để hở hoặc đã đặt logic 1; nếu cho $\overline{LT} = 0$ logic thì cả 7 đầu ra từ a đến g đều ở mức thấp và cả 7 thanh hiển thị đều phải sáng, chứng tỏ hiển thị là tốt.

- Đầu $\overline{BI}/\overline{RBO}$ có thể đóng vai trò như một đầu vào làm trắng $\overline{BI}$ (Blanking Input). Khi cho $\overline{BI}/\overline{RBO} = 0$ logic thì cả 7 thanh hiển thị đều tắt, không phụ thuộc mức logic ở các đầu vào khác. Đầu $\overline{BI}/\overline{RBO}$ cũng có thể đóng vai trò như một đầu ra làm trắng số thập phân 0 (Ripple Blanking Output: RBO), sẽ trình bày dưới.

- Đầu $\overline{RBI}$ tích cực thấp, gọi là đầu vào làm trắng số thập phân 0 (Ripple Blanking Input). Khi cho $\overline{RBI} = 0$ logic, $\overline{LT} = 1$ logic và cửa vào DCBA = 0000, cả 7 đầu ra từ a đến g đều ở mức logic cao. Nghĩa là 7 thanh hiển thị đều tắt thay vì cho việc lẽ ra phải hiển thị số 0. Đầu $\overline{BI}/\overline{RBO}$ lúc này cũng mang giá trị 0 logic và nó đóng vai trò đầu ra làm trắng số thập phân 0 (Ripple Blanking Output) trong hiển thị số thập phân nhiều chữ số như hình 11.4.



Hình 11.4: Hiển thị số thập phân nhiều chữ số dùng 7446

Tóm lại, khi sử dụng riêng lẻ một IC 7446, 7447 để làm hiển thị một chữ số thập phân như hình 11.3 ta phải đặt đầu LT ở 1 logic, các đầu $\overline{RBI}$, $\overline{BI}/\overline{RBO}$ hoặc để hở, hoặc đặt ở 1 logic.

Hai đầu $\overline{RBI}$, $\overline{BI}/\overline{RBO}$ chỉ sử dụng khi dùng nhiều IC 7446 để thành lập mạch điều khiển hiển thị số thập phân nhiều chữ số (hình 11.4). Quy tắc nối như sau: Ở hai chữ số thập phân có trọng số cực đại và cực tiểu ta đặt 0 logic vào đầu $\overline{RBI}$. Đầu ra $\overline{BI}/\overline{RBO}$ của số thập phân có trọng số cao/thấp nối với đầu vào $\overline{RBI}$ của chữ số thập phân có trọng số thấp hơn/cao hơn liền kề (tùy thuộc chúng ở phần nguyên hay phần lẻ của số thập phân). Các đầu $\overline{RBI}$ và $\overline{BI}/\overline{RBO}$ của chữ số thập phân hàng đơn vị (trọng số 10^0) và hàng 1/10 (trọng số 10^{-1}) thì luôn để hở hoặc đặt mức logic cao (+5V). Hình 11.4 nối theo quy tắc vừa nêu. Nhờ các đầu $\overline{RBI}$ và $\overline{BI}/\overline{RBO}$ mà mạch hình 11.4 sẽ không hiển thị các số 0 không cần thiết trong dãy số hiển thị. Ví dụ, với mã BCD ở đầu vào, đáng lẽ mạch phải hiển thị số 007.050, thì nó chỉ làm hiển thị số 7.05. Ta thấy mạch đã làm trắng ba số 0 đáng lẽ phải hiển thị. Kết quả đọc trở nên rõ ràng hơn. Cần chú ý ở đây là mức logic ở đầu $\overline{RBI}$ hoàn toàn không có ảnh hưởng tới việc làm hiển thị các số từ 1 đến 9.

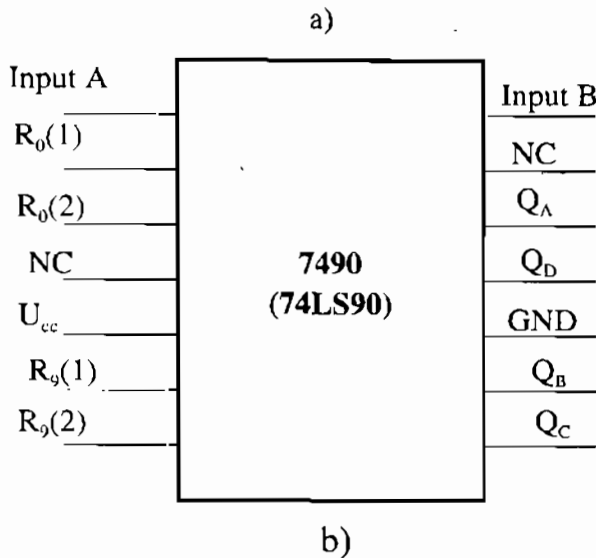
3. Các vi mạch đếm

3.1. Vi mạch 7490 (74LS90)

Đây là vi mạch chứa bộ đếm thập phân gồm 1 bộ đếm 2 (dùng FF-A) và 1 bộ đếm 5 (dùng 3 FF B,C,D theo thứ tự B là bit có trọng số thấp nhất và D là bit có trọng số cao nhất). Q_A , Q_B , Q_C , Q_D là các đầu ra của FF.

Vi mạch này gồm 14 chân trong đó có chân không dùng đến ký hiệu là NC (No Connection).

Vào thiết lập				Ra			
$R_0(1)$	$R_0(2)$	$R_9(1)$	$R_9(2)$	Q_D	Q_C	Q_B	Q_A
H	H	L	X	L	L	L	L
H	H	X	L	L	L	L	L
X	X	H	H	H	L	L	H
X	L	X	L				đếm
L	X	L	X				đếm
L	X	X	L				đếm
X	L	L	X				đếm



Hình 11.5: Vi mạch đếm thập phân 7490 và 74LS90

4 chân thiết lập: R₀(1), R₀(2), R₉(1), R₉(2). Khi R₀(1) = R₀(2) = mức cao (H) thì bộ đếm được xoá về không và các đầu ra ở mức thấp.

R₉(1) = R₉(2) = H thì mạch được thiết lập ở trạng thái 9. Hai đầu này nhất thiết phải có một đầu ở mức thấp và 2 đầu R₀(1) và R₀(2) không cùng ở mức cao thì mạch mới đếm.

- A_{in} dùng để đưa tín hiệu vào đếm ở bộ đếm 2 (FF-A).
- B_{in} dùng để đưa tín hiệu vào đếm ở bộ đếm 5 (3 FF B,C,D).

Khi nối Q_A vào B_{in} và đưa tín hiệu đếm vào A_{in} thì bộ đếm sẽ đếm 10.

Hình 11.5 à bảng chân lý và minh hoạ sơ đồ chân của 7490.

II. MỘT SỐ SƠ ĐỒ THỰC TẾ

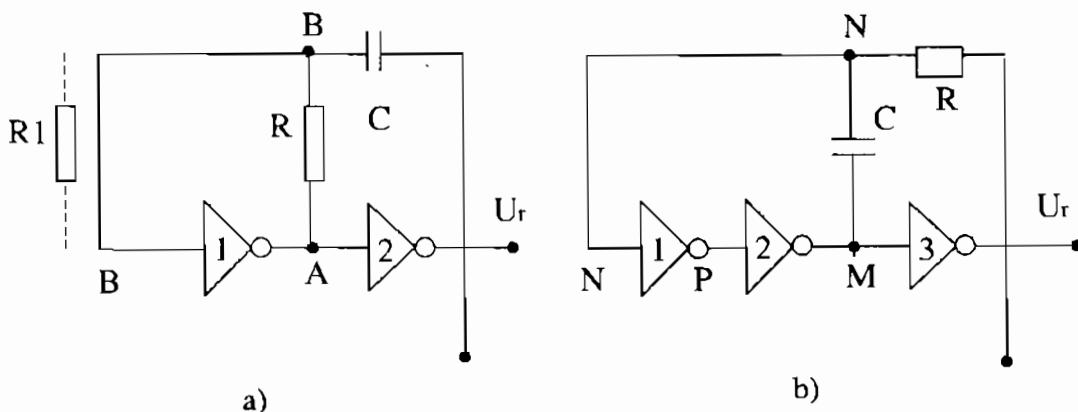
1. Mạch tạo xung nhịp

Dùng các phần tử NAND/NOR để tạo thành những mạch đa hài tự dao động (đa hài phiếm định). Khi cấp nguồn, mạch phát ra dãy xung lặp vuông góc với độ chính xác khá cao. Ta chia các mạch này thành hai nhóm: Các mạch không điều khiển và các mạch có điều khiển.

1.1. Mạch đa hài phiếm định không điều khiển

Là mạch *liên tục* phát ra dãy xung vuông góc với chu kỳ lặp xác định, không có sự điều khiển giới hạn khoảng thời gian phát xung. Hình 11.6 là sơ

đồ cơ bản của mạch, gồm hai phần tử Đảo nối tăng và mạch định thời R-C. Các phần tử đảo trong sơ đồ thường do phần tử NAND/NOR tạo thành.



Hình 11.6: Mạch đa hài phiếm định không điều khiển

Xét sự làm việc của mạch hình 11.6.a. Giả sử ở thời điểm bắt đầu khảo sát, U_r , ở mức cao $H \approx +U =$ điện áp nguồn cung cấp, điểm A ở mức thấp $L \approx 0V$ và điểm B ở mức cao H; tụ C không tích điện và được nạp theo chiều từ $U_r = H \rightarrow C \rightarrow R \rightarrow A$. Điện áp ở điểm B sẽ giảm dần từ mức cao H theo sự giảm của dòng nạp cho tụ. Khi điểm B đạt giá trị ngưỡng lật $U_T \approx H/2 \approx +U/2$, thì phần tử đảo số 1 lật trạng thái và điểm A nhảy lên mức cao H, điện áp ra U_r tụt xuống mức thấp $L \approx 0V$. Tụ C lúc đó cũng nhanh chóng phóng điện qua mạch ở cửa vào của phần tử 1. Điện áp ở điểm B vì vậy cũng tụt ngay xuống mức thấp L. Tụ C trở nên không tích điện và bắt đầu được nạp theo chiều ngược lại, từ $A = H \rightarrow R \rightarrow C \rightarrow U_r = L$. Điện áp ở điểm B vì vậy tăng dần từ mức thấp L theo sự tăng của điện áp trên tụ. Khi B đạt ngưỡng lật U_T thì phần tử đảo số 1 lại lật trạng thái và điểm A lại chuyển từ H sang L, U_r nhảy từ L lên H. Tụ C một lần nữa nhanh chóng phóng hết điện qua mạch vào của phần tử 1 và $B = H$. Tụ lại bắt đầu được nạp theo chiều từ $U_r = H \rightarrow C \rightarrow R \rightarrow A = L$. Quá trình tiếp tục lặp lại như nhau. Kết quả mạch cho ta điện áp ra U_r , là một dãy xung vuông góc đối xứng với chu kỳ lặp:

$$T \approx 1,4 RC$$

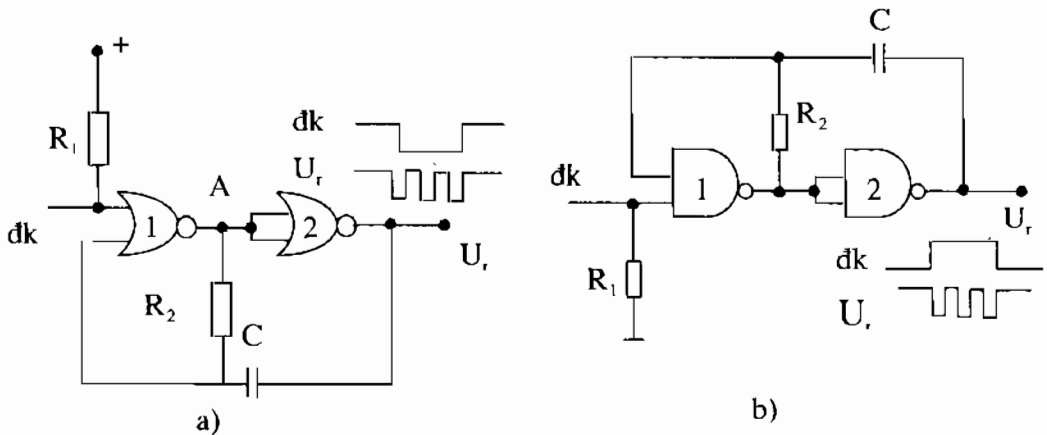
Và biên độ xấp xỉ điện áp nguồn cung cấp $+U$. Nếu chọn tụ C từ vài nF đến vài μF (không dùng tụ hoá) và điện trở R trong dải từ vài K Ω đến vài chục M Ω , tần số của dãy xung ra có thể nằm trong dải từ vài Hz đến khoảng 1 MHz. Điện áp nguồn cung cấp cho IC họ CMOS có thể lấy từ 3 ÷ 18V.

Nhược điểm của mạch hình 11.6.a là dãy xung vuông góc nhận được không hoàn toàn đối xứng; tần số của dãy xung thay đổi theo điện áp nguồn cung cấp và theo các “chip” IC khác nhau được sử dụng. Khi điện áp nguồn cung cấp tăng 10% thì tần số dãy xung phát ra giảm khoảng 0,8%. Để giảm tối đa các nhược điểm của mạch hình 11.6.a, ở đường về cửa vào của phân tử số 1, người ta đặt thêm các điện trở R_1 (trên sơ đồ hình 11.6.a có vẽ điện trở R_1 ở bên cạnh với nét đứt khúc, ý nói có thể chèn thêm vào). Trị số của R_1 chọn rất lớn hơn R (ví dụ chọn $R_1 = 10 R$)

Mạch đa hài phiếm định hai tầng ở hình 11.6.a còn có một nhược điểm quan trọng là sườn lên và xuống của các xung không trơn, tồn tại các gai xung nhiều ở hai sườn xung. Để loại bỏ nhược điểm này, ta dùng mạch đa hài phiếm định ba tầng ở hình 11.6.b. Điện áp ra U_r của mạch là một dãy xung vuông góc đối xứng, không có gai xung nhiều ở hai sườn xung. Hoạt động của mạch như sau: Giả sử ở thời điểm ban đầu khảo sát, điện áp ra U_r ở mức thấp $L \approx 0V$ và tụ C không tích điện, điểm M và N có điện áp nằm ở mức cao $H \approx +U$, ($+U$ là điện áp nguồn cung cấp), điện áp điểm P là ở mức thấp L . Tụ C bắt đầu được nạp theo chiều từ $M =$ mức cao $H \rightarrow C \rightarrow N \rightarrow R \rightarrow U_r = L$. Điện áp ở điểm N giảm dần từ mức cao H theo sự giảm của dòng nạp. Khi điện áp điểm N giảm tới giá trị ngưỡng lật U_{Tr} , thì phân tử đảo số 1 lật trạng thái và đầu ra P của nó chuyển ngay lên mức cao H , còn đầu M lật sang mức thấp L và U_r nhảy lên mức cao H . Tụ C nhanh chóng phóng điện hết qua cửa vào của phân tử số 1 và điểm N cũng chuyển ngay xuống mức thấp L . Bây giờ, tụ C lại bắt đầu nạp theo chiều ngược lại, từ $U_r = H \rightarrow R \rightarrow C \rightarrow M = L$. Điện áp ở điểm N tăng dần từ mức thấp $L \approx 0V$, cho tới khi đạt ngưỡng lật U_{Tr} thì phân tử 1 lại lật trạng thái, điểm P chuyển xuống mức thấp L và điểm M nhảy lên mức cao H . Điện áp ra U_r chuyển nhanh xuống mức thấp L . Tụ C mau chóng phóng hết điện qua cửa vào của phân tử số 1 và N cũng ở mức cao H . Quá trình tiếp diễn lặp lại như đã trình bày ở trên. Kết quả U_r là một dãy xung vuông góc đối xứng không có gai xung nhiều ở hai sườn xung.

1.2. Mạch đa hài phiếm định có điều khiển

Nó là mạch chỉ có thể phát ra dãy xung vuông góc trong một thời khoảng xác định mà ta có thể điều khiển được. Để có loại mạch này, ta dùng mạch đa hài phiếm định không điều khiển ở hình 11.6, với dãy xung ra U_r được dẫn tới một cổng VÀ hay HOẶC. Dùng một xung có độ rộng t_x để điều khiển mở cổng cho dãy xung U_r đi qua. Sau thời gian t_x thì cổng đóng và dãy xung bị chặn.



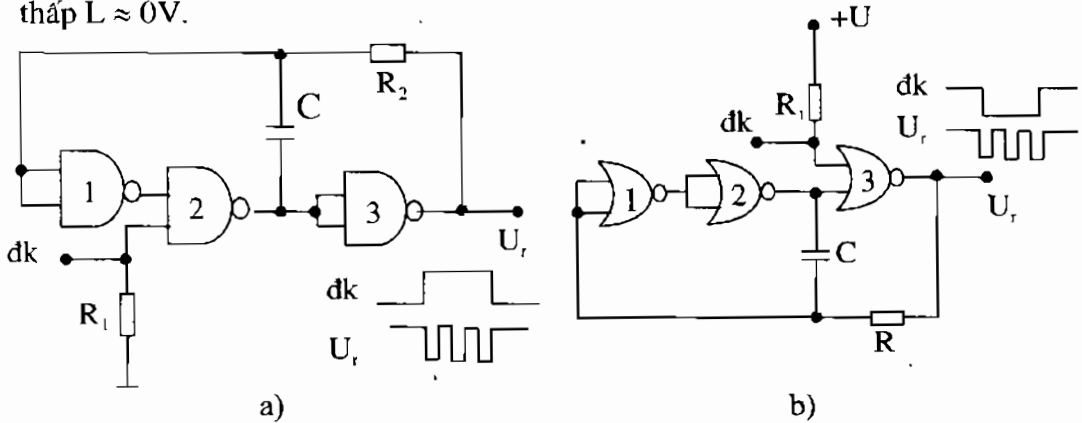
Hình 11.7: Mạch đa hài phiếm định hai tầng, có điều khiển

Tuy nhiên, từ tính chất là các phân tử NAND/NOR cũng có thể sử dụng như một cổng logic, ta có thể thành lập các mạch đa hài phiếm định hai tầng có điều khiển như ở hình 11.7, trong đó dk là tín hiệu điều khiển để mạch làm việc, U_r là dãy xung ra.

Xét mạch hình 11.7.a. Khi tín hiệu điều khiển dk ở mức cao H (hoặc khi không có tín hiệu điều khiển), phân tử NOR số 1 trở thành cổng logic ở trạng thái đóng. Điểm A nằm cố định ở mức điện áp thấp $L \approx 0V$ và U_r nằm ổn định ở mức cao $H \approx +U$. Tụ C được nạp tới mức cao H. Mạch không làm việc. Khi tín hiệu điều khiển dk ở mức thấp L, cổng 1 mở và trở thành phân tử đảo. Mạch lúc đó có cấu trúc hoàn toàn giống mạch hình 11.6.a và nó phát ra dãy xung vuông góc đối xứng. Vậy mạch chỉ làm việc cho ra dãy xung vuông góc trong thời gian khi $dk = L$. Một cách tương tự, ta dễ dàng nhận thấy mạch hình 11.7.b chỉ làm việc và phát xung khi tín hiệu điều khiển $dk = H$. Khi $dk = L$, mạch dừng và U_r nằm cố định ở mức L.

Mạch đa hài phiếm định hai tầng có điều khiển ở trên phát ra dãy xung vuông góc đối xứng không đẹp, ở hai sườn xung có các gai xung nhiễu. Để khắc phục nhược điểm này, người ta xây dựng các mạch đa hài phiếm định 3 tầng có điều khiển (hình 11.8). Về mặt cấu trúc, những mạch này hoàn toàn dựa trên cấu trúc mạch 3 tầng không điều khiển ở hình 11.6.b, nhưng có một tầng trong mạch (phần tử số 2 ở hình 11.8.a và phần tử số 3 ở hình 11.8.b) được sử dụng như một cổng logic với tín hiệu điều khiển dk để đóng/mở cổng. Khi $dk =$ mức cao H (đối với hình 11.8.a) hoặc $dk =$ mức thấp L (đối với hình

11.8.b) thì mạch hoàn toàn trở thành mạch hình 11.6.b và nó phát ra dãy xung vuông góc mong muốn. Ngược lại, khi $dk = L$ (đối với hình 11.8.a) hay $dk = H$ (với hình 11.8.b) thì mạch không hoạt động, điện áp ra U_r nằm cố định ở mức thấp $L \approx 0V$.



Hình 11.8: Mạch đa hài phiếm định 3 tầng, có điều khiển

1.3. Vi mạch định thời 555 (timer 555)

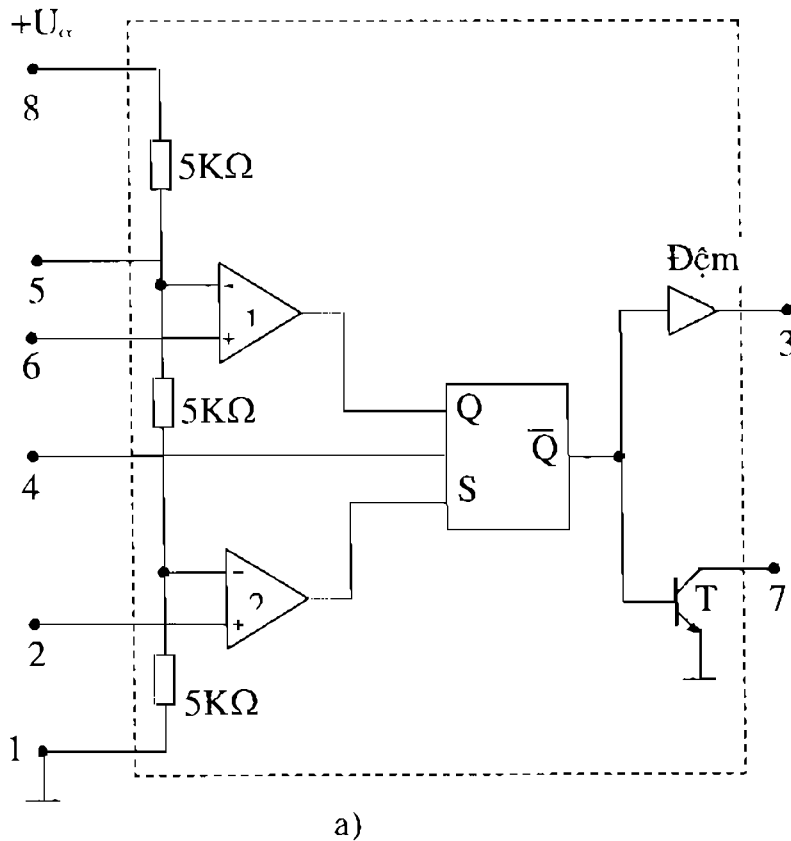
Timer 555 là một vi mạch định thời rất thông dụng. Nó có thể sử dụng theo nhiều chức năng: làm mạch đa hài đơn ổn hoặc phiếm định, để tạo một xung vuông đơn hay một dãy xung vuông góc lặp lại, hoặc một dãy xung tam giác. Thời gian định thời có thể thay đổi từ vài μs đến vài trăm giây (s) nhờ một mạch R-C đơn giản, với độ chính xác điển hình là $\pm 1\%$.

1.3.1. Khái niệm về mạch định thời 555

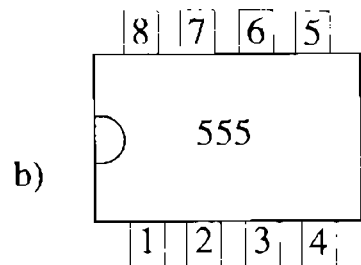
Có hai loại mạch định thời thông dụng nhất là Timer 555 chế tạo theo công nghệ transistor lưỡng cực (BJT) và Timer 7555 chế tạo theo công nghệ CMOS. Điện áp cung cấp cho IC 555 là $5 \rightarrow 15V$, cho IC 7555 là $2 \rightarrow 18V$. Công suất tiêu thụ và dòng cung cấp ở IC 7555 cũng nhỏ hơn nhiều so với ở IC 555. Với nguồn $+15V$, IC 555 yêu cầu dòng cung cấp cỡ $10mA$. Dòng đầu ra cực đại của IC 555 có thể đạt tới $200mA$ (ở IC 7555 chỉ bằng khoảng $1/2$).

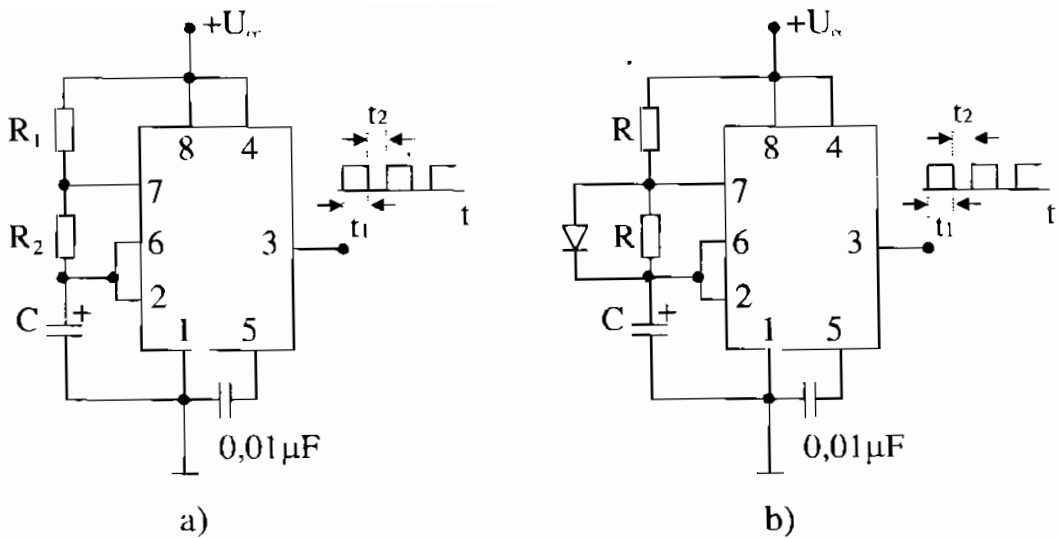
Hình 11.9.a, b tương ứng là sơ đồ khối chức năng và vỏ kiểu hai hàng chân song song của IC 555. Ở đây ta chỉ xét vai trò của tám chân (1-8) trên vỏ IC ở hình 11.9.b: Chân số 8 để đặt nguồn cung cấp $U_{cc} = 5 \rightarrow 15V$. Chân số 1 là chân nối "mát". Chân số 2 là đầu vào kích khởi (trigger), dùng để đặt xung kích thích bên ngoài khi mạch làm việc ở chế độ đa hài đơn ổn. Chân số 3 là đầu ra của IC. Chân số 4 là chân xóa (Reset); nó có thể điều khiển xóa điện áp

đầu ra khi điện áp đặt vào chân này từ 0,7V trở xuống. Vì vậy, để có thể phát ra xung ở đầu ra, chân số 4 phải đặt ở mức cao H. Chân số 5 là chân điện áp điều khiển (Control Voltage). Ta có thể đưa một điện áp ngoài vào chân này để làm thay đổi việc định thời của mạch, nghĩa là làm thay đổi tần số dãy xung phát ra. Khi không được sử dụng thì chân 5 nối xuống “mất” thông qua một tụ khoảng $0,01\mu\text{F}$. Chân 6 là chân điện áp ngưỡng (Threshold). Chân 7 là chân phóng điện (Discharge).



Hình 11.9: Sơ đồ khối chức năng
và vỏ IC 555





Hình 11.10: Dùng IC 555 làm mạch đa hài phiếm định
và làm mạch phát dãy xung răng cưa

1.3.2. Chế độ làm việc như mạch đa hài phiếm định của IC 555

Hình 11.10. a là sơ đồ nối của IC 555 để nó làm việc như một mạch đa hài phiếm định. Chân ra số 3 sẽ phát ra một dãy xung vuông góc lặp lại. Từ hình

11.10. a ta thấy: Thời gian tồn tại xung t_1 (độ rộng xung) phụ thuộc vào tốc độ nạp của tụ C từ nguồn cung cấp, nghĩa là tỷ lệ với hằng số thời gian nạp $\tau_n = (R_1 + R_2)C$, ta có:

$$t_1 = (R_1 + R_2) C \ln 2 \approx 0,7 (R_1 + R_2) C \quad (11-1)$$

Thời gian không có xung t_2 (thời gian nghỉ) phụ thuộc sự phóng điện của tụ C qua chân phóng điện số 7, nghĩa là tỷ lệ với hằng số thời gian phóng $\tau_p = R_2 C$ và:

$$t_2 = R_2 C \ln 2 \approx 0,7 R_2 C \quad (11-2)$$

Vậy, tần số của dãy xung ở đầu ra:

$$f = 1/T = 1/(t_1 + t_2) = 1,44/((R_1 + 2R_2)C) \quad (11-3)$$

Muốn nhận được dãy xung vuông góc đối xứng, nghĩa là $t_1 = t_2 = T/2$, ta có thể làm theo hai cách sau:

*Chọn $R_1 \ll R_2$. Lúc đó theo (11-1) và (11-2) có thể coi $t_1 \approx t_2 = 0,7 R_2 C$. Tuy nhiên không thể chọn R_1 quá nhỏ được, ví dụ $R_1 = 100\Omega$, vì khi đó dòng đi

từ nguồn $+U_{CC}$ vào chân 7 khi transistor T dẫn (hình 11.10.a) là $+U_{CC}/R_1$ sẽ quá lớn, phá hỏng transistor.

*Sử dụng thêm một diod mắc song song với R_2 và chọn $R_1 = R_2 = R$ (hình 11.10.b). Ở hình 11.10.b, đường nạp cho tụ C từ nguồn $+U_{CC}$ có đi qua diod; điện trở R nối song song với diod khi đó coi như bị ngắn mạch và hằng số thời gian của mạch nạp $\tau_n = RC$. Khi tụ C phóng điện vào chân 7 thì nó không thể phóng qua diod (vì mắc ngược) và hằng số thời gian của mạch phóng $\tau_p = RC$. Ta thấy $\tau_n = \tau_p$ nên:

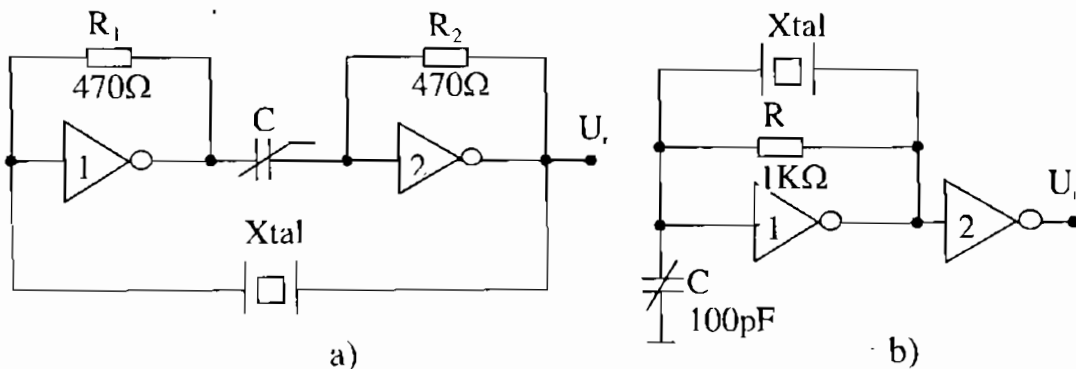
$$t_1 = t_2 \approx 0,7 RC$$

Dãy xung ở đầu ra là đối xứng, với tần số:

$$f = 1/T = 1/(t_1 + t_2) = 1/1,4RC \quad (11-4)$$

1.4. Mạch tạo dãy xung vuông góc dùng tinh thể thạch anh

Khi cần dãy xung vuông góc với độ chính xác tần số rất cao, tần số dãy xung cũng cao (thường là 1 MHz $\rightarrow$ 10 MHz), người ta hay dùng mạch phát xung thạch anh. Ở đây, tinh thể thạch anh áp điện được sử dụng như mạch cộng hưởng cơ điện với độ chính xác rất cao.



Hình 11.11: Các mạch phát xung vuông góc dùng tinh thể thạch anh

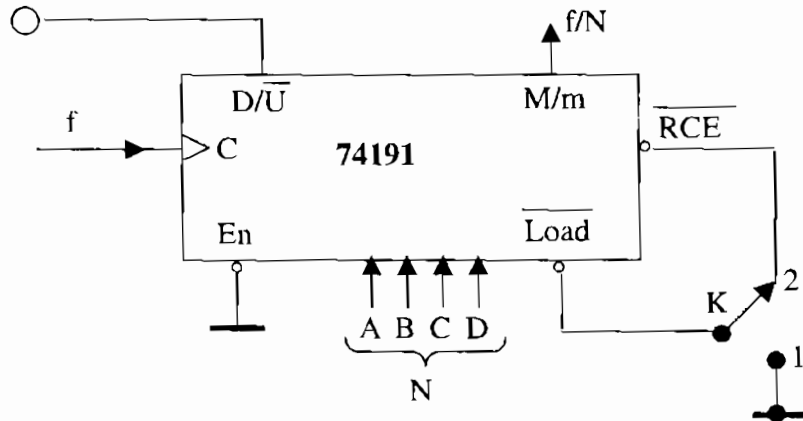
Hình 11.11a dùng 2 phân tử đảo họ TTL của IC 74LS04 để tạo thành mạch phát xung thạch anh có tần số ở trong dải từ vài trăm kHz đến 10MHz. Tinh thể thạch anh loại hoạt động kiểu nối tiếp. Tụ C khoảng vài nF và phải chỉnh định theo tần số của dãy xung ra.

Hình 11.11.b dùng hai phân tử đảo của vi mạch họ CMOS (IC 4049B) để hợp thành với tinh thể thạch anh hoạt động kiểu song song, tạo thành mạch phát dãy xung vuông góc với tần số cỡ MHz.

2. Các mạch sử dụng bộ đếm

2.1. Mạch chia tần số có thể lập trình

* Hình 11-12 giới thiệu mạch chia tần số với số chia N có thể lập trình sử dụng bộ đếm là IC 74191.

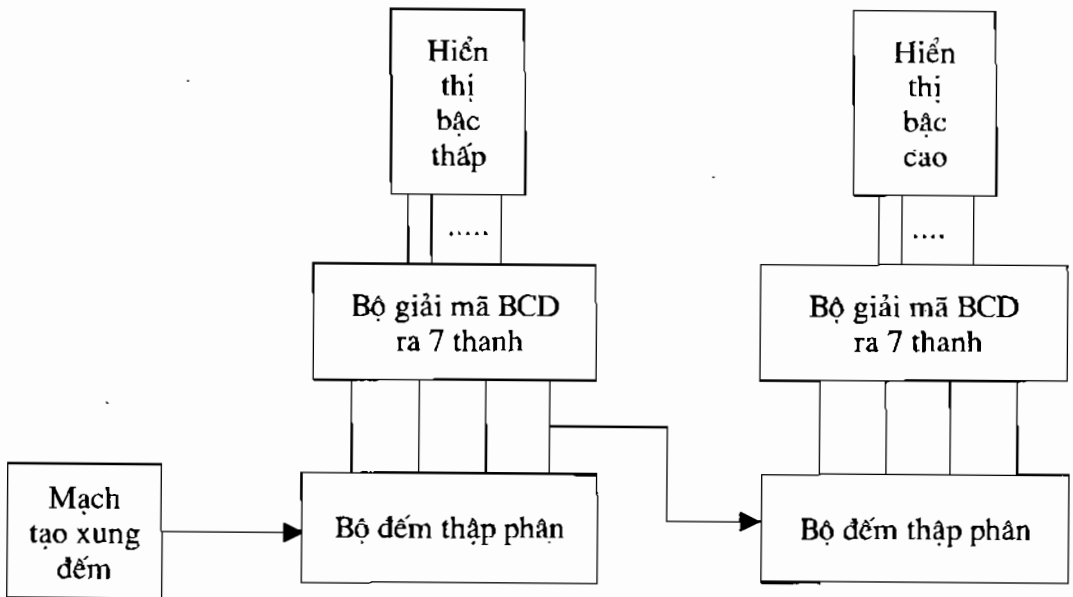


Hình 11.12: Mạch chia tần số có thể lập trình

Xét hoạt động của mạch: Đặt số chia N vào các đầu vào số liệu A, B, C, D. Cho bộ đếm làm việc theo cách đếm ngược ($D/U = 1$ logic). Dây xung tần số f cần chia dẫn vào đầu C. Đầu M/m chính là đầu ra mạch chia tần. Khởi động đóng K sang vị trí 1, số N được nạp vào bộ đếm. Đầu RCE ở 1 logic. Đóng K sang vị trí 2, bộ đếm đếm ngược từ số N. Đếm được N xung thì nội dung bộ đếm về số 0, đầu RCE tụt xuống 0 logic và số N lại được nạp vào bộ đếm. Đầu ra M/m phát ra một xung và RCE lại nhảy lên mức cao. Bộ đếm tiếp tục một chu trình đếm mới. Quá trình cứ liên tục tiếp diễn như trên. Ta thấy, cứ N xung vào bộ đếm thì đầu M/m cho một xung ra. Vậy dây xung ở đầu ra M/m có tần số là f/N .

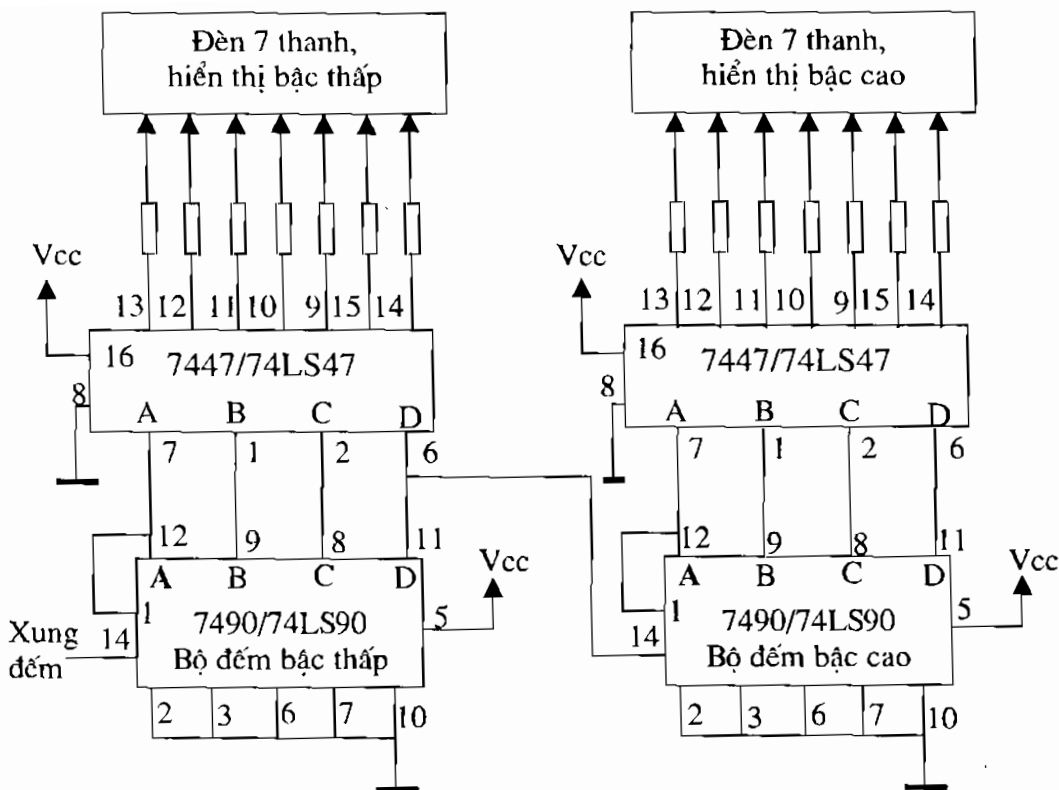
2.2. Mạch đếm thập phân hiển thị hai chữ số

Trong mạch đếm này sử dụng hai bộ đếm thập phân. Xung đếm được đưa vào mạch đếm bậc thấp. Mạch đếm bậc cao nhận được xung đưa vào đầu vào đếm mỗi khi mạch đếm bậc thấp chuyển trạng thái từ 1010 về 0000. Nguyên tắc mạch đếm hiển thị 2 chữ số thập phân được thể hiện ở hình 11.13. Kết quả đếm đưa qua bộ giải mã BCD ra 7 thanh và đưa vào kích thích cho đèn 7 thanh hiển thị.



Hình 11.13: Sơ đồ khối mạch đếm 2 số thập phân

Hình 11.14 vẽ sơ đồ mạch đếm hiển thị kết quả đếm là hai chữ số thập phân. Mạch sử dụng vi mạch 7490 hoặc 74LS90 làm bộ đếm thập phân. Bộ đếm sẽ thực hiện đếm từ $(00)_{10}$ đến $(99)_{10}$, vì vậy các chân 2, 3, 6, 7 đều nối xuống mass. Khi bộ đếm bậc thấp đạt được giá trị $(1001)_2$, xung đếm thứ 10 đưa vào, bộ đếm này chuyển trạng thái thành $(1010)_2$, trạng thái bộ đếm bị xoá ngay về bằng (0000) . Với việc chuyển trạng thái như vậy, chân D của bộ đếm bậc thấp tạo ra tín hiệu xung âm đưa vào đầu vào đếm của bộ đếm bậc cao. Bộ đếm bậc cao chuyển trạng thái. Như vậy, bộ đếm bậc thấp cứ đếm được 10 xung thì bộ đếm bậc cao lại đếm được một xung. Đầu ra của mỗi bộ đếm nối vào một bộ giải mã BCD ra 7 thanh 7447 hoặc 74LS47 (loại dùng cho đèn hiển thị 7 thanh AC). Trạng thái đếm của mạch luôn được thể hiện trên 2 đèn hiển thị 7 thanh khi bộ đếm hoạt động.



Hình 11.14: Mạch đếm thập phân hiển thị 2 chữ số

3. Bộ ghi dịch vạn năng 7495

3.1. Sơ đồ

Bộ ghi dịch 4 bit, dùng các RS-FF mắc theo kiểu D-FF. Bộ ghi dịch này có thể đưa thông tin vào nối tiếp hoặc song song. Nó cũng cho phép lấy thông tin ra nối tiếp hoặc song song.

Như vậy, bộ ghi dịch trên có 4 khả năng vào ra thông tin (vào nối tiếp ra nối tiếp, vào nối tiếp ra song song, vào song song ra nối tiếp, vào song song ra song song).

Khi đưa thông tin vào song song, các bit dữ liệu được đặt ở các đầu vào A, B, C, D. Lúc đó $MC = 1$ và khi có xung Ck2 thì dữ liệu được đưa vào đồng thời.

Muốn lấy thông tin ra nối tiếp, đặt $MC = 0$ và dùng xung nhịp Ck1.

Muốn lấy thông tin ra song song thì từ các đầu ra A, B, C, D được nối tới đầu nhận dữ liệu của một mạch khác thông qua một số cổng logic.

Ngoài ra, với cách nối mạch như hình 11.15a bộ ghi dịch có thể dịch thông tin theo cả hai hướng phải và trái.

Câu hỏi và bài tập

1. Trình bày chức năng và đặc điểm của các vi mạch số thông dụng.
2. Trình bày về đèn hiển thị 7 thanh 7446/7447 và cách sử dụng để hiển thị nhiều số thập phân.
3. Trình bày IC 7490/74LS90.
4. Phân tích hoạt động của các mạch tạo xung nhịp.
5. Phân tích mạch chia tần số.
6. Vẽ sơ đồ và phân tích mạch đếm hiển thị 4 chữ số thập phân.
7. Trình bày về bộ ghi dịch 7495.

TÀI LIỆU THAM KHẢO

1. **Kỹ thuật điện tử số**, Đỗ Thanh Hải và Trương Trọng Tuấn, Nhà xuất bản Thanh niên, 2002
2. **Giáo trình kỹ thuật xung số**, Lương Ngọc Hải, Nhà xuất bản Giáo dục, 2004.
3. **Kỹ thuật số thực hành**, Huỳnh Đắc Thắng, Nhà xuất bản Khoa học và kỹ thuật, 1995.
4. **Cơ sở kỹ thuật điện tử số**, Vũ Đức Thọ dịch, Nhà xuất bản Giáo dục, 2003.
5. **Sổ tay tra cứu IC số**, Dương Minh Trí, Nhà xuất bản Khoa học và kỹ thuật, 1999.
6. **Kỹ thuật số**, Nguyễn Thuý Vân, Nhà xuất bản Khoa học và kỹ thuật, 1995.

MỤC LỤC

<i>Lời giới thiệu</i>	3
<i>Lời nói đầu</i>	5
<i>Bài mở đầu</i>	7
Chương 1. HỆ THỐNG ĐẾM VÀ MÃ	
I. Biểu diễn số trong các hệ thống đếm	9
II. Chuyển đổi số giữa các hệ đếm	11
III. Phép toán cộng, trừ trong các hệ đếm 2, 8, 16	18
IV. Các mã thông dụng	21
Chương 2. ĐẠI SỐ LOGIC	
I. Các khái niệm và phép toán logic	30
II. Các cổng logic cơ bản	36
III. Các phương pháp biểu diễn hàm logic	48
IV. Khái niệm về tối thiểu hoá và các phương pháp tối thiểu	59
V. Phương pháp tối thiểu hoá bằng công thức	61
VI. Phương pháp tối thiểu hoá bằng bảng Karnaugh	63
VII. Mạch tổ hợp	67
Chương 3. CÁC BỘ CỘNG, TRỪ VÀ SO SÁNH	
I. Bộ cộng hai số nhị phân	74
II. Bộ trừ hai số nhị phân	80
III. Bộ so sánh hai số nhị phân	84
Chương 4. BỘ DỒN KÊNH VÀ BỘ PHÂN KÊNH	
I. Khái quát chung	91
II. Bộ dồn kênh	93
III. Bộ phân kênh	96
IV. Ứng dụng của bộ dồn kênh và phân kênh	98
Chương 5. CÁC BỘ MÃ HOÁ VÀ GIẢI MÃ	
I. Khái niệm chung	104
II. Bộ mã hoá	105
III. Bộ giải mã n đầu vào, 2 ⁿ đầu ra	109
IV. Bộ giải mã BCD ra thập phân	111

V. Bộ giải mã BCD ra 7 thanh	114
Chương 6. CÁC PHẦN TỬ NHỚ CƠ BẢN	
I. Khái niệm về mạch dây và FF	119
II. RS-FF	122
III. T-FF	127
IV. JK-FF	129
V. D-FF	134
VI. Mạch chốt	136
Chương 7. BỘ ĐẾM	
I. Khái niệm về bộ đếm	139
II. Bộ đếm tiến không đồng bộ Mod tám	144
III. Bộ đếm lùi không đồng bộ	148
IV. Bộ đếm tiến lùi không đồng bộ	150
V. Bộ đếm Mod 10 không đồng bộ	154
VI. Bộ đếm đồng bộ Mod 16	160
VII. Bộ đếm thiết lập trước	163
Chương 8. BỘ GHI DỊCH	
I. Khái niệm	167
II. Bộ ghi dịch nạp thông tin vào nối tiếp	168
III. Bộ ghi dịch nạp thông tin vào song song	171
Chương 9. BỘ NHỚ	
I. Khái niệm về bộ nhớ	175
II. Bộ nhớ RAM	176
III. Các bộ nhớ ROM	182
Chương 10. CHUYỂN ĐỔI TÍN HIỆU	
I. Khái niệm về tín hiệu tương tự và tín hiệu số	189
II. Mạch bộ chuyển đổi tín hiệu số sang tín hiệu tương tự	190
III. Mạch bộ chuyển đổi tín hiệu tương tự sang tín hiệu số	200
Chương 11. CÁC VI MẠCH SỐ THÔNG DỤNG VÀ MỘT SỐ SƠ ĐỒ THỰC TẾ	
I. Các vi mạch số thông dụng	208
II. Một số sơ đồ thực tế	214
Tài liệu tham khảo	227

NHÀ XUẤT BẢN HÀ NỘI
4 - TỐNG DUY TÂN, QUẬN HOÀN KIẾM, HÀ NỘI
ĐT: (04) 8252916, 8257063 - FAX: (04) 8257063

GIÁO TRÌNH
KỸ THUẬT SỐ VÀ MẠCH LOGIC
NHÀ XUẤT BẢN HÀ NỘI - 2006

Chịu trách nhiệm xuất bản
NGUYỄN KHẮC OÁNH

Biên tập
PHẠM QUỐC TUẤN

Bìa
TRẦN QUANG

Kỹ thuật vi tính
LƯU NGỌC TRÂM

Sửa bản in
PHẠM QUỐC TUẤN
LÊ XUÂN THỌ

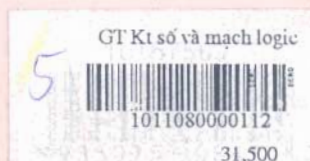
In 980c khổ 17x24cm. Tại cơ sở 2-Công ty cổ phần in 15.

GP XB số: 154-2006/CXB/586 GT-15/HN cấp ngày 28-02-2006.

In xong và nộp lưu chiểu quý IV năm 2006.

BỘ GIÁO TRÌNH XUẤT BẢN NĂM 2006
KHỐI TRƯỜNG TRUNG HỌC ĐIỆN TỬ - ĐIỆN LẠNH

1. LÝ THUYẾT MẠNG VÀ HỆ THỐNG ĐIỀU HÒA MẠNG
2. THIẾT KẾ HỆ THỐNG LẠNH
3. THỰC HÀNH BẢO TRÌ HỆ THỐNG MÁY TÍNH
4. THỰC HÀNH SỬA CHỮA MÁY LẠNH
5. ĐỒNG BỘ VÀ BÁO HIỆU TRONG MẠNG VIỄN THÔNG
6. TỔ CHỨC MẠNG VÀ DỊCH VỤ VIỄN THÔNG
7. THIẾT BỊ ĐẦU CUỐI
8. NHIỆT KỸ THUẬT
9. MÀN HÌNH VÀ CARD ĐIỀU KHIỂN MÀN HÌNH
10. ĐO LƯỜNG KỸ THUẬT LẠNH
11. THỰC HÀNH KỸ THUẬT SỐ
12. THỰC HÀNH ĐIỆN TỬ CÔNG NGHIỆP
13. CẤU TRÚC MÁY TÍNH
14. LÝ THUYẾT BẢO TRÌ HỆ THỐNG MÁY TÍNH
15. KỸ THUẬT VI XỬ LÝ
16. KỸ THUẬT SỐ VÀ MẠCH LOGÍC
17. KỸ THUẬT THÔNG TIN QUANG
18. THỰC HÀNH LINUX
19. THỰC HÀNH MẠNG
20. KỸ THUẬT ĐIỀU HÒA KHÔNG KHÍ
21. THỰC HÀNH GIA CÔNG LẮP ĐẶT ĐƯỜNG ỐNG
22. MÁY VÀ THIẾT BỊ LẠNH
23. THỰC HÀNH MÀN HÌNH MÁY TÍNH
24. THỰC HÀNH VIỄN THÔNG CHUYÊN NGÀNH



Giá: 31.500đ